NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

ADAPTIVE ARMA LATTICE FILTER BASED ON A GENERALIZED MULLIS-ROBERTS CRITERION

by

Donald W. Mennecke

June 1988

Thesis Advisor: Murali Tummala

Unclassified

security classification of this page

REPORT DOCUMENTATION PAGE

1a Report Security Classification Unclassified	1b Restrictive Markings
2a Security Classification Authority	3 Distribution Availability of Report Approved for public release; distribution is unlimited.
2b Declassification Downgrading Schedule	
4 Performing Organization Report Number(s)	5 Monitoring Organization Report Number(s)
6a Name of Performing Organization Naval Postgraduate School — 6b Office Symbol *(if applicable)* 32	7a Name of Monitoring Organization Naval Postgraduate School
6c Address *(city, state, and ZIP code)* Monterey, CA 93943-5000	7b Address *(city, state, and ZIP code)* Monterey, CA 93943-5000
8a Name of Funding Sponsoring Organization — 8b Office Symbol *(if applicable)*	9 Procurement Instrument Identification Number
8c Address *(city, state, and ZIP code)*	10 Source of Funding Numbers Program Element No · Project No · Task No · Work Unit Accession No

11 Title *(include security classification)* ADAPTIVE ARMA LATTICE FILTER BASED ON A GENERALIZED MULLIS-ROBERTS CRITERION

12 Personal Author(s) Donald W. Mennecke

13a Type of Report	13b Time Covered	14 Date of Report *(year, month, day)*	15 Page Count
Master's Thesis	From To	June 1988	83

16 Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 Cosati Codes			18 Subject Terms *(continue on reverse if necessary and identify by block number)*
Field	Group	Subgroup	Adaptive, Filter, ARMA, Lattice, Modeling

19 Abstract *(continue on reverse if necessary and identify by block number)*

In this thesis, an adaptive lattice algorithm is derived for an ARMA digital lattice filter, whose parameters are estimated using a generalized Mullis-Roberts criterion for parameter estimation. Design of the ARMA lattice filter based on this generalized criterion is studied as is the accuracy of the parameter estimation algorithm used in its design. Application of the derived lattice algorithm to system identification modeling is demonstrated through computer simulation of various system identification problems.

20 Distribution Availability of Abstract ☒ unclassified unlimited ☐ same as report ☐ DTIC users	21 Abstract Security Classification Unclassified
22a Name of Responsible Individual Murali Tummala	22b Telephone *(include Area code)* (408) 646-2645 — 22c Office Symbol 62Tu

DD FORM 1473,84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete — security classification of this page

Unclassified

Approved for public release; distribution is unlimited.

Adaptive ARMA Lattice Filter Based on a Generalized Mullis-Roberts Criterion

by

Donald W. Mennecke
Lieutenant, United States Navy
B.A., State University of New York At Buffalo, 1981

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 1988

Author: Donald W Mennecke
Donald W. Mennecke

Approved by: Murali Tummala
Murali Tummala, Thesis Advisor

Roberto Cristi, Second Reader

John P. Powers, Chairman,
Department of Electrical and Computer Engineering

G E Schacher
Gordon E. Schacher,
Dean of Science and Engineering

ABSTRACT

In this thesis, an adaptive lattice algorithm is derived for an ARMA digital lattice filter, whose parameters are estimated using a generalized Mullis-Roberts criterion for parameter estimation. Design of the ARMA lattice filter based on this generalized criterion is studied as is the accuracy of the parameter estimation algorithm used in its design. Application of the derived lattice algorithm to system identification modeling is demonstrated through computer simulation of various system identification problems.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

Digital signal processing is a field which is rapidly expanding due to advances in modern technology. Essential to this field are digital filters. Modeling these filters constitutes much of the effort involved in digital signal processing. The filters provide a transfer function which describes the relationship between filter input and output. Autoregressive (AR), moving average (MA) and combination autoregressive moving average (ARMA) models are widely used to represent the transfer function of a digital filter. A filter transfer function is commonly described in direct form. This form is a ratio of two polynomials, usually of the form,

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + \cdots + b_t z^{-t}}{1 + a_1 z^{-1} + \cdots + a_s z^{-s}} \tag{1.1}$$

The above equation describes an ARMA model of order (s,t) where s is the order of the denominator and t the order of the numerator. The a_s parameters form the autoregressive portion of the ARMA model. The b_t parameters form the moving average portion of the ARMA model. If all the autoregressive parameters are zero, then the filter transfer function H(z) is strictly a moving average process of order t. If all the moving average parameters are zero except for b_0 equal to one, then the filter transfer function is strictly an autoregressive process of order s.

A. OBJECTIVES OF THE THESIS

Fundamental to the design of digital filters is estimation of AR, MA or ARMA parameters. Accurate and efficient parameter estimation has been the subject in much of the related digital signal processing literature [Refs. 1 ,2,3]. The first objective of this thesis is to confirm the proposed ARMA parameter estimation algorithm of [Ref. 4: pp. 619-621], which leads to the design of a new ARMA digital lattice filter. The proposed algorithm is a generalization of the Mullis-Roberts criterion for parameter estimation known as the modified least squares problem [Ref. 5: pp. 227-228]. The algorithm uses two recursive formula to estimate the parameters. One is an AR recursive formula which estimates ARMA parametes as the AR order is increased by one. The other is an MA recursive formula which estimates ARMA parameters as the MA order is increased by one. This algorithm is unique in that it allows for the design of an ARMA model with arbitrary AR and MA orders with no dependency of an AR model on an MA model or vice versa. The ARMA digital filter designed from the proposed ARMA parameter estimation algorithm is in the form of a lattice structure. Lattice realizations of filters are widely used and provide excellent analysis of prediction errors [Refs. 6 : pp. 165-168,7]. Gray and Markel developed an algorithm which produces lattice realizations of filters from the direct form [Ref. 8].

The second objective of the thesis is to make the proposed ARMA digital lattice filter of [Ref. 4: p. 662] adaptive. An adaptive lattice filter is one in which the lattice coefficients are automatically adjusted by an adaptive algorithm to yield the optimum filter design. The adaptive lattice algorithm derived in this thesis is based on the widely used least mean square (LMS) algorithm. Adaptive filters have many applications [Ref. 9: pp. 7-31] including,

1. System identification.
2. Digital representation of speech.
3. Adaptive auotoregressive spectrum analysis.
4. Adaptive detection of a signal in noise of unknown statistics.
5. Echo cancellation.
6. Adaptive line enhancment.
7. Adaptive beamforming.

The need for an adaptive filter is made apparent by considering a filter of fixed design which is optimized for given input conditions. In practice, the complete range of input conditions may not be known or could change from time to time. A filter of fixed design would not produce optimum results under these conditions. An adaptive filter, which yields optimum results given changing input conditions, will give superior performance to one of fixed design.

The last objective is to analyze convergence properties of the derived adaptive lattice algorithm. This is accomplished by computer implementation of the adaptive algorithm. The output of a known transfer function is compared to the output of the adaptive lattice filter given a common input. Plots of the error between the two outputs and lattice coefficient convergence are obtained.

B. ORGANIZATION OF THE THESIS

Chapter II is designed to present the ARMA parameter estimation algorithm and ARMA digital lattice filter proposed in [Ref. 4: pp. 617-628]. Computer simulation of the algorithm was performed and results are shown. A brief review of the Mullis-Roberts criterion is provided to establish a reference for expanding this criterion to the proposed ARMA parameter estimation algorithm. An adaptive lattice algorithm is derived in Chapter III which makes the proposed ARMA digital lattice filter adaptive. The adaptive lattice algorithm is efficient and accurate. Chapter IV contains experimental results which show convergence aspects of the adaptive lattice algorithm when applied to ARMA lattice filters. Conclusions about the proposed ARMA parameter estimation algorithm as well as the derived adaptive algorithm are discussed.

II. ARMA DIGITAL LATTICE FILTER

In this chapter, we will review the Mullis-Roberts criterion for solving linear approximation problems and introduce analysis equations of the ARMA digital lattice filter. The criterion used in the formulation of the ARMA digital lattice filter is a generalized form of the Mullis-Roberts criterion [Ref. 5: pp. 227-228], which has been given as a modified least mean square problem for ARMA parameter estimation.

A. MULLIS-ROBERTS CRITERION

The Mullis-Roberts criterion evolved from considering second order statistics in conjuction with first order information about a process to obtain filter approximations. Consider the bounded impulse response sequence $h = \{h_0, h_1, \dots\}$ containing first order information about the filter h having a frequency response function,

$$H(e^{j\omega}) = \sum_{k=0}^{\infty} h_k e^{-j\omega k} \tag{2.1}$$

Let $\{ u_t \}$ be a zero-mean, unit-variance, white-noise sequence and $\{ y_t \}$ be the output process corresponding to the input $\{ u_t \}$, then we have the following convolutional relationship given by,

$$y_t = \sum_{k=0}^{\infty} h_k u_{t-k} \tag{2.2}$$

Second order information about the filter h is obtained from the autocorrelation sequence $\{ r_k \}$ given as

$$r_k = E(y_t y_{t+k}) = \sum_{i=0}^{\infty} h_i h_{k+i} \tag{2.3}$$

From equations (2.1) and (2.2), the second order interpolation problem is to find a lowest order recursive filter which matches the data $\{ h_0, \dots, h_m, r_0, \dots, r_m \}$, where h_i represents the first order information and r_j the second order statistics.

Let us now consider the case where only first order information about a process is known. That is, given an impulse response sequence $\{ h_0, h_1, \ldots \}$, we want to find a recursive filter of the form,

$$\hat{H}(z) = \sum_{k=0}^{\infty} \hat{h}_k z^{-k} = \frac{q_0 + q_1 z^{-1} + \cdots + q_m z^{-m}}{1 + a_1 z^{-1} + \cdots + a_n z^{-n}} \tag{2.4}$$

which approximates $H(z)$ and therefore the impulse response sequence $\{ h_0, h_1, \ldots \}$. We also desire the frequency response $\hat{H}(e^{j\omega})$ to approximate the desired response $H(e^{j\omega})$. Suppose that $\hat{H}(e^{j\omega})$ is chosen such that it minimizes the integral squared error,

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} |H(e^{j\omega}) - \hat{H}(e^{j\omega})|^2 d\omega \quad = \quad \|h - \hat{h}\|^2 \tag{2.5}$$

Using the Parseval relation, we can obtain an alternative definition of the approximation error,

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} |H(e^{j\omega}) - \hat{H}(e^{j\omega})|^2 d\omega \quad = \quad \|h - \hat{h}\|^2 \quad = \sum_{k=0}^{\infty} (h_k - \hat{h}_k)^2 \tag{2.6}$$

If the filters $H(z)$ and $\hat{H}(z)$ are driven by the same white noise source, equation (2.6) describes the output error between the filters which we write as,

$$\|h - \hat{h}\|^2 \quad = \quad E(y_t - \hat{y}_t)^2 \quad = \quad E(e_t)^2 \tag{2.7}$$

where y_t and $\hat{y}_t$ are the outputs of the respective filters when driven by the same white noise source as in (2.2). Minimizing (2.7) is a nonlinear programming problem requiring the entire impulse response sequence. As a result, computational efforts for obtaining a solution are inefficient. A modification to the problem was introduced [Ref. 5: pp. 227-228] which considered a cost function that is quadratic in the coefficients of the recursive filter given by (2.4). The modification is described as follows. Let

$$Q(z) = z^{N}(q_0 + q_1 z^{-1} + \cdots + q_m z^{-m}) \tag{2.8}$$

and

$$A(z) = z^{N}(1 + a_1 z^{-1} + \cdots + a_n z^{-n}) \tag{2.9}$$

be the numerator and denominator polynomials, respectively, in (2.4), where $N = \max(m,n)$. The task now is to find coefficients which minimize the quadratic form,

$$E(e_t)^2 = \frac{1}{2\pi}\int_{-\pi}^{\pi} |H(e^{j\omega})A(e^{j\omega}) - Q(e^{j\omega})|^2 d\omega \tag{2.10}$$

This is a standard quadratic minimization problem whose integral can be expressed in terms of the coefficients of polynomials A(z) and Q(z) and the data $\{h_0, \ldots, h_m, r_0, \ldots, r_m\}$, relating to the filter $H(z)$. This problem is shown in Figure 1 and equation (2.10) is known as the Mullis-Roberts criterion.

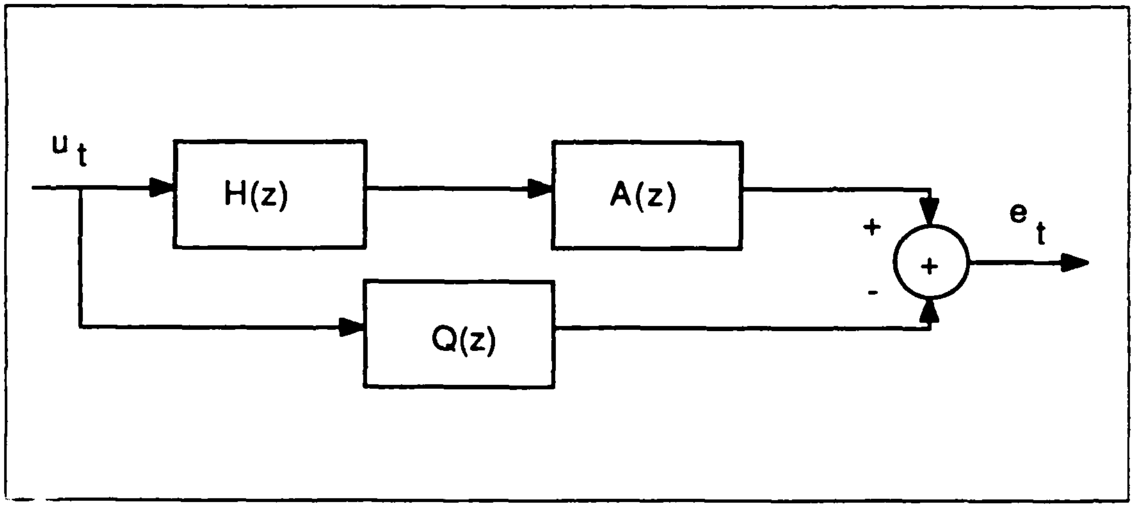

Figure 1. Modified least squares problem

B. GENERALIZED MULLIS-ROBERTS CRITERION

In order to define the new criterion used for ARMA parameter estimation we consider the following transfer function with input sequence $\{ x(k), k = 1,2, \ldots \}$ and output sequence $\{ y(k), k = 1,2, \ldots \}$ written as,

$$H(z) = \frac{H_y(z)}{H_x(z)} \tag{2.11}$$

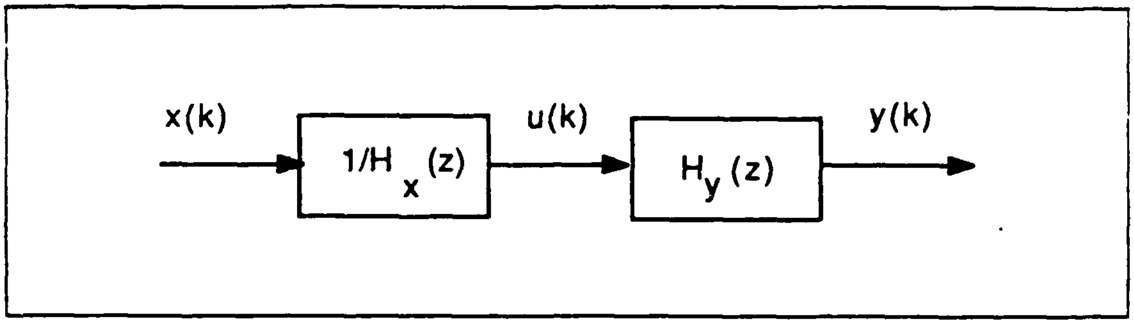

Figure 2. Equivalent input/output model

where $H_y(z)$ and $H_x(z)$ are reference polynomials which we desire to model. An equivalent model is shown in Figure 2 where $u(k)$ is an intermediate signal, and the realization is similar to that of the direct form realization II [Ref. 10: p. 151]. Let the intermediate sequence $\{ u(k), k = 1,2, \ldots \}$ be a zero-mean white gaussian process. The model of Figure 2 can then be transformed into the model of Figure 3 with $u(k)$ as the common input to both transfer functions $H_y(z)$ and $H_x(z)$.

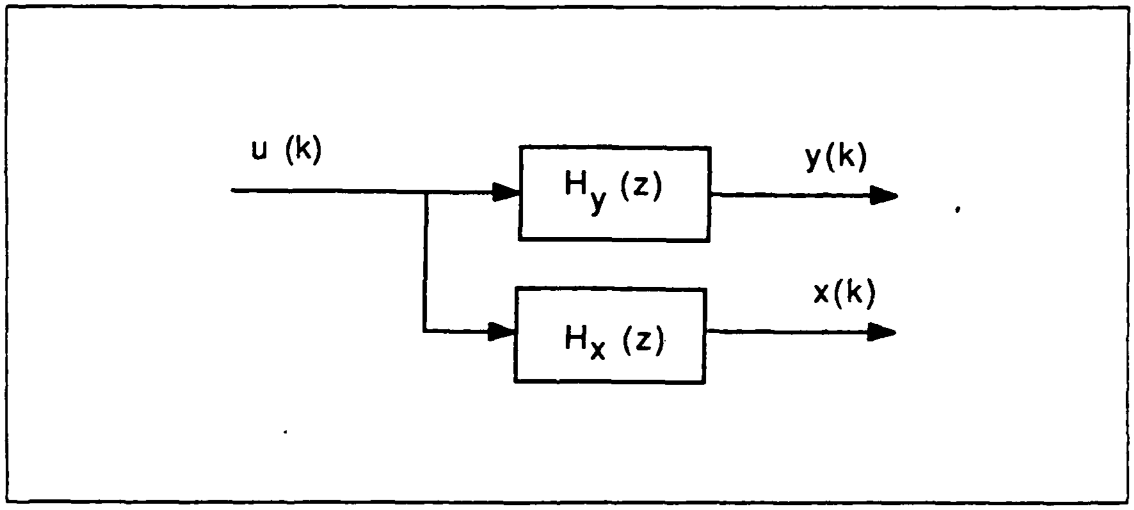

Figure 3. Transformed model

Note that we earlier defined transfer functions $H_y(z)$ and $H_x(z)$ as polynomials of the reference model. For each transfer function an estimation polynomial is defined such that

$$A(z) = 1 + a_1 z^{-1} + \cdots + a_s z^{-s} \quad \textit{estimates} \quad H_x(z) \tag{2.12}$$

$$B(z) = b_0 + b_1 z^{-1} + \cdots + b_t z^{-t} \quad \textit{estimates} \quad H_y(z) \tag{2.13}$$

where a_i and b_j, $(i = 1,2, \ldots , s)$ and $(j = 1,2, \ldots , t)$, are the AR and MA parameters, respectively, of the combined ARMA model formed by A(z) and B(z). This refined least squares problem is shown in Figure 4 and is a generalized form of the Mullis-Roberts criterion. If the reference model polynomial $H_x(z)$ in Figure 4 is equal to unity, we obtain the Mullis-Roberts criterion shown in Figure 1. Therefore, by including reference polynomial $H_x(z)$, the new criterion for ARMA parameter estimation becomes,

$$E_{s,t} = \frac{\sigma_u^2}{2\pi j} \oint |H_y(z)A(z) - H_x(z)B(z)|^2 \frac{dz}{z} \tag{2.14}$$

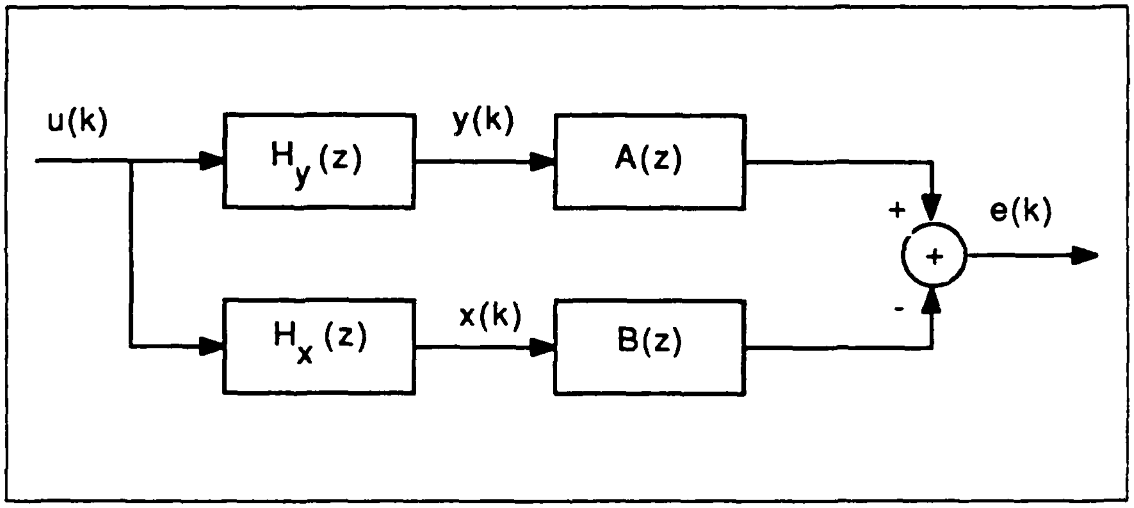

Figure 4. Refined least squares problem

Minimizing $E_{s,t}$ is accomplished by calculating the coefficients of A(z) and B(z) which mininize $e(k)^2$ of Figure 4. Another form of eq (2.14) is obtained by applying Parseval's theorem and is expressed as,

$$E_{s,t} = E[(A(z)y(k) - B(z)x(k))^2] \tag{2.15}$$

which is obvious from Figure 4. The coefficients a_i, $(i = 1, \ldots , s)$, and b_j, $(j = 1, \ldots , t)$, which minimize $E_{s,t}$ can then be calculated using the normal equations for ARMA parameter estimation. In order to obtain the normal equations for the problem in (2.15), let us define the following:

$$\mathbf{a}_{s,t} = [a_1 \ldots a_s] \quad \text{and} \quad \mathbf{b}_{s,t} = [b_1 \ldots b_t] \tag{2.16}$$

are the vectors of AR and MA parameters, respectively,

$$\mathbf{h}_{s,t}(k) = [\ y(k) \ldots y(k-s) \quad -x(k) \ldots -x(k-t)\ ] \tag{2.17}$$

is the data vector consisting of both input and output data elements and

$$\mathbf{R}_{s,t} = E[\mathbf{h}_{s,t}(k)^T\, \mathbf{h}_{s,t}(k)] \tag{2.18}$$

is the data autocorrelation matrix. The criterion is to minimize the mean squared error

$$\begin{aligned} E_{s,t} = E\left[\ e^2(k)\ \right] &= E\Big[\ \big[\ [\ 1\ \ \mathbf{a}_{s,t}\ \ b_0\ \ \mathbf{b}_{s,t}\ ]\ \mathbf{h}_{s,t}^T\ \big]^2 \Big] \\ &= E\Big[\ \big[\ y(k) + [\ \mathbf{a}_{s,t}\ \ b_0\ \ \mathbf{b}_{s,t}\ ]\ \mathbf{h}_{s,t}^T\ \big]^2 \Big] \end{aligned} \tag{2.19}$$

Now following the standard calculus of variation optimization procedure for minimizing $E_{s,t}$, [Ref. 11: pp. 100-110], yields the normal equations in the matrix form,

$$[\ 1\ \ \mathbf{a}_{s,t}\ \ b_0\ \ \mathbf{b}_{s,t}\ ]\ \mathbf{R}_{s,t} = [\ \min E_{s,t}\ \ 0\ \ 0\ \ 0\ ] \tag{2.20}$$

It is interesting to note that if $E_{s,t}$ in equation (2.14) is zero, then we have the following equality,

$$H(z) = \frac{H_y(z)}{H_x(z)} = \frac{B(z)}{A(z)} \tag{2.21}$$

so the estimate for the total reference model H(z) is the ratio of B(z), the MA part and A(z), the AR part of an estimated ARMA model.

C. ARMA PARAMETER ESTIMATION

Now that the criterion for ARMA parameter estimation has been established, the solution method to estimate the ARMA model parameters to minimize equation (2.14) is considered. Let $x(k)$ and $y(k)$ be the input and output signals, respectively, of the estimated ARMA model. Using a difference equation representation, this process is described by

$$y(k) = -\sum_{j=1}^{s} a_j \ y(k-j) + \sum_{i=0}^{t} b_i \ x(k-i) \tag{2.22}$$

For these input and output signals we define four estimation, or prediction, models as follows. The forward estimation signal for $x(k)$ is defined as,

$$\hat{x}_f(k) = -\sum_{i=1}^{t} b_i^x \ x(k-i) + \sum_{j=1}^{s} a_j^x \ y(k-j) \tag{2.23}$$

where b_i^x and a_j^x are the corresponding estimation parameters. The forward esitmation signal for $y(k)$ is similarly defined as,

$$\hat{y}_f(k) = -\sum_{j=1}^{s} a_j^y \ y(k-j) + \sum_{i=1}^{t} b_i^y \ x(k-i) \tag{2.24}$$

The backward estimation errors for $x(k-t)$ and $y(k-s)$ are then given by,

$$\hat{x}_b(k-t) = -\sum_{i=0}^{t-1} b_i^g \ x(k-i) + \sum_{j=0}^{s-1} a_j^g \ y(k-j) \tag{2.25}$$

$$\hat{y}_b(k-s) = -\sum_{j=0}^{s-1} a_j^d \ y(k-j) + \sum_{i=0}^{t-1} b_i^d \ x(k-i) \tag{2.26}$$

where the superscripts g and d indicate the backward estimation parameters for x and y, respectively.

From these estimation models, we can now obtain the four prediction errors at any given time k. These errors are expressed as differences between the predicted value and actual value of an input or output signal, namely,

$$v_{s,t}^x(k) = -x(k) + \hat{x}_f(k) \tag{2.27}$$

$$v_{s,t}^y(k) = y(k) - \hat{y}_f(k) \tag{2.28}$$

$$g_{s,t}(k) = x(k-t) - \hat{x}_b(k-t) \tag{2.29}$$

$$d_{s,t}(k) = y(k-s) - \hat{y}_b(k-s) \tag{2.30}$$

We now use the vector notation to simplify the expressions for prediction errors. In the following, the forward error elements corresponding to both x and y form a vector $\mathbf{v}_{s,t}(k)$, given by

$$\mathbf{v}_{s,t}(k) = [\ -v_{s,t}^x(k) \quad v_{s,t}^y(k)] = \mathbf{h}_{s,t}(k)\,\mathbf{C}_{s,t}^T \tag{2.31}$$

and the backward error vectors are given by

$$g_{s,t}(k) = -\,\mathbf{h}_{s,t}(k)\,\mathbf{G}_{s,t}^T \tag{2.32}$$

$$d_{s,t}(k) = \mathbf{h}_{s,t}(k)\,\mathbf{D}_{s,t}^T \tag{2.33}$$

where $\mathbf{C}_{s,t}$ and $\mathbf{G}_{s,t}$ and $\mathbf{D}_{s,t}$ are the forward estimation parameter matrix and backward estimation parameter vectors, respectively, defined as

$$\mathbf{C}_{s,t} = \begin{bmatrix} 0 & a_1^x & \dots & a_s^x & 1 & b_1^x & \dots & b_t^x \\ 1 & a_1^y & \dots & a_s^y & 0 & b_1^y & \dots & b_t^y \end{bmatrix} \tag{2.34}$$

$$\mathbf{G}_{s,t} = [a_0^g \quad \dots \quad a_{s-1}^g \quad 0 \quad b_0^g \quad \dots \quad b_{t-1}^g \quad 1] \tag{2.35}$$

$$\mathbf{D}_{s,t} = [a_0^d \quad \dots \quad a_{s-1}^d \quad 1 \quad b_0^d \quad \dots \quad b_{t-1}^d \quad 0] \tag{2.36}$$

It can be shown that the prediction errors satisfy some orthogonal conditions. These conditions are similar to those found in AR modeling problems [Ref. 6 : pp. 116-121]. We now list the orthogonality conditions for the ARMA formulation in discussion without proof as the following:

$$E[\ v_{s,t}^x(k) \quad y(k-j)\ ] = 0 \qquad E[\ v_{s,t}^y(k) \quad y(k-j)\ ] = 0$$

$$E[\ v_{s,t}^x(k) \quad x(k-i)\ ] = 0 \qquad E[\ v_{s,t}^y(k) \quad x(k-i)\ ] = 0$$

$$E[g_{s,t}(k-1) \quad y(k-j)] = 0 \tag{3.37}$$

$$E[g_{s,t}(k-1) \quad x(k-i)] = 0$$

$$E[d_{s,t}(k-1) \quad y(k-j)] = 0$$

$$E[d_{s,t}(k-1) \;\; x(k-i)] = 0$$

where $i = 1,2, \ldots, t$ and $j = 1,2, \ldots, s$.

In Section B we have obtained a set of normal equations in terms of $\mathbf{R}_{s,t}$ and the ARMA estimation parameters. In this section, we have defined four sets of forward and backward estimation parameters and established some orthogonal relationships. In what follows, we derive a set of equations which relates the coefficients of the estimated ARMA model with those of the forward estimation parameter matrix $\mathbf{C}_{s,t}$. Consider the expected value of the forward prediction error, $\mathbf{v}_{s,t}(k)$ and the data $\mathbf{h}_{s,t}(k)$. Since the prediction error is orthogonal to all past samples of data $y(k-j)$, $x(k-i)$ but not to $y(k)$ or $x(k)$ as listed in (2.37), the result is a matrix which is defined as

$$E\left[\, \mathbf{v}_{s,t}(k)^T \; \mathbf{h}_{s,t}(k) \,\right] = \begin{bmatrix} \xi_1 & \mathbf{0} & \xi_2 & \mathbf{0} \\ \xi_3 & \mathbf{0} & \xi_4 & \mathbf{0} \end{bmatrix} \tag{2.38}$$

where

$$\xi_1 = -E\left[\, v^x_{s,t}(k)\, y(k) \,\right] \;=\; -E\left[\, v^x_{s,t}(k)\; v^y_{s,t}(k) \,\right] = \; E^{xy}_{s,t} \tag{2.39}$$

is the crosscorrelation between the forward prediction errors of x and y at lag zero.

$$\xi_2 = \; E\left[\, v^x_{s,t}(k)\, x(k) \,\right] = \; E\left[\, (v^x_{s,t}(k))^2 \,\right] = E^x_{s,t} \tag{2.40}$$

is the forward prediction error power of $x(k)$

$$\xi_3 = \; E\left[\, v^y_{s,t}(k)\, y(k) \,\right] = \; E\left[\, (v^y_{s,t}(k))^2 \,\right] = E^y_{s,t} \tag{2.41}$$

is the forward prediction error power of $y(k)$ and

$$\xi_4 = -E\left[\, v^y_{s,t}(k)\, x(k) \,\right] = -E\left[\, v^y_{s,t}(k)\; v^x_{s,t}(k) \,\right] = E^{xy}_{s,t} \tag{2.42}$$

is the crosscorrelation between the forward prediction errors of y and x at lag zero. In another interpretation, the left hand side of equation (2.38) can be written as,

$$E\left[\, \mathbf{v}_{s,t}(k)^T \mathbf{h}_{s,t}(k) \,\right] = \; E\left[\, \mathbf{C}_{s,t}\, \mathbf{h}_{s,t}(k)^T\, \mathbf{h}_{s,t}(k) \,\right] = \mathbf{C}_{s,t}\, \mathbf{R}_{s,t} \tag{2.43}$$

and we have

$$\mathbf{C}_{s,t}\,\mathbf{R}_{s,t} = \begin{bmatrix} E^{xy}_{s,t} & 0 & E^{x}_{s,t} & 0 \\ E^{y}_{s,t} & 0 & E^{xy}_{s,t} & 0 \end{bmatrix} \tag{2.44}$$

from equation (2.38).

A similar approach for both backward prediction errors yields the following

$$\begin{bmatrix} \mathbf{G}_{s,t} \\ \mathbf{D}_{s,t} \end{bmatrix} \mathbf{R}_{s,t} = \begin{bmatrix} 0 & E^{gd}_{s,t} & 0 & E^{g}_{s,t} \\ 0 & E^{d}_{s,t} & 0 & E^{gd}_{s,t} \end{bmatrix} \tag{2.45}$$

In order to express the coefficients of the ARMA estimation model in terms of the coefficients of the parameter estimation matrix $\mathbf{C}_{s,t}$ and parameter vectors $\mathbf{G}_{s,t}$ and $\mathbf{D}_{s,t}$, consider the combination of the normal equation (2.20) and the parameter estimation matrix and vectors. From equation (2.44) the normal equation for ARMA parameter estimation may be written as

$$\begin{bmatrix} 0 & \mathbf{a}^{x}_{s,t} & 1 & \mathbf{b}^{x}_{s,t} \\ 1 & \mathbf{a}^{y}_{s,t} & 0 & \mathbf{b}^{y}_{s,t} \end{bmatrix} \mathbf{R}_{s,t} = \begin{bmatrix} E^{xy}_{s,t} & 0 & E^{x}_{s,t} & 0 \\ E^{y}_{s,t} & 0 & E^{xy}_{s,t} & 0 \end{bmatrix} \tag{2.46}$$

Combining equation (2.46) with (2.20) we obtain

$$\begin{bmatrix} 1 & \mathbf{a}_{s,t} & b_0 & \mathbf{b}_{s,t} \\ 0 & \mathbf{a}^{x}_{s,t} & 1 & \mathbf{b}^{x}_{s,t} \\ 1 & \mathbf{a}^{y}_{s,t} & 0 & \mathbf{b}^{y}_{s,t} \end{bmatrix} \mathbf{R}_{s,t} = \begin{bmatrix} E_{s,t}\text{min} & 0 & 0 & 0 \\ E^{xy}_{s,t} & 0 & E^{x}_{s,t} & 0 \\ E^{y}_{s,t} & 0 & E^{xy}_{s,t} & 0 \end{bmatrix} \tag{2.47}$$

In equation (2.47) multiply row 2 by $\frac{E^{xy}_{s,t}}{E^{x}_{s,t}}$ then subtract row 2 from row 3 and equate the result to row 1. We then obtain the following relations

$$\mathbf{a}_{s,t} = \mathbf{a}^{y}_{s,t} - \mathbf{a}^{x}_{s,t}\,\frac{E^{xy}_{s,t}}{E^{x}_{s,t}} \tag{2.48}$$

$$b_0 = \frac{E^{xy}_{s,t}}{E^{x}_{s,t}} \tag{2.49}$$

$$\mathbf{b}_{s,t} = \mathbf{b}^{y}_{s,t} - \mathbf{b}^{x}_{s,t}\,\frac{E^{xy}_{s,t}}{E^{x}_{s,t}} \tag{2.50}$$

and,

$$E_{s,t}\min = E^{y}_{s,t} - \frac{(E^{xy}_{s,t})^2}{E^{x}_{s,t}} \tag{2.51}$$

Calculation of the ARMA estimation model coefficients requires knowledge of the estimation parameters $\mathbf{a}^{x}_{s,t}$, $\mathbf{a}^{y}_{s,t}$, $\mathbf{b}^{x}_{s,t}$, $\mathbf{b}^{y}_{s,t}$, b_0 and the values $E^{x}_{s,t}$, $E^{y}_{s,t}$, $E^{xy}_{s,t}$. Recursive formulas calculate the estimation parameters as the AR or MA order of the estimation model increases by one. Given the parameters of the ARMA estimation model of order $(s\ ,\ t)$ these formulas calculate the $(s+1\ ,\ t)$ or $(s\ ,\ t+1)$ ARMA parameters. For a comprehensive derivation of these recursive formulae see [Ref. 4: pp. 619-621]. The AR-type recursive formula for the forward parameter estimation matrix and backward estimation parameter vectors is

$$\begin{aligned}
\mathbf{C}_{s+1,t} &= \mathbf{C}_{s,t}\,\mathbf{I}_1 + \mathbf{u}_1^T\,\mathbf{D}_{s,t}\,\mathbf{I}_2 \\
\mathbf{G}_{s+1,t} &= [\ \mathbf{G}_{s,t} + u_2\,\mathbf{D}_{s,t}\ ]\ \mathbf{I}_1 \\
\mathbf{D}_{s+1,t} &= \mathbf{D}_{s,t}\,\mathbf{I}_2 + [\ \mathbf{u}_3\,\mathbf{C}_{s,t} + u_4\,\mathbf{G}_{s,t} + u_5\,\mathbf{D}_{s,t}\ ]\ \mathbf{I}_1
\end{aligned} \tag{2.52}$$

where

$$\begin{aligned}
\mathbf{u}_1 &= -\,(E^{d}_{s,t})^{-1}\ [\ \tau_1\ \ \tau_2\ ] \\
u_2 &= \frac{-E^{g\,d}_{s,t}}{E^{d}_{s,t}} \\
\mathbf{u}_3 &= -\ [\ \tau_1\ \ \tau_2\ ]\ E^{-1}_{s,t} \\
u_4 &= \frac{(E^{g\,d}_{s,t}\,\tau_4 - E^{d}_{s,t}\,\tau 3)}{\left[\ (E^{g\,d}_{s,t})^2 - E^{d}_{s,t}\,E^{g}_{s,t}\ \right]} \\
u_5 &= u_4\,u_2
\end{aligned} \tag{2.53}$$

and the (s+ 1,t) prediction error powers are recursively calulated as follows

$$\begin{aligned}
E_{s+1,t} &= E_{s,t} + \mathbf{u}_1^T\ [\ \tau_1\ \ \tau_2\ ] \\
E^{g\,d}_{s+1,t} &= \tau_3 + u_2\,\tau_4 \\
E^{d}_{s+1,t} &= E^{d}_{s,t} + [\ \tau_1\ \ \tau_2\ ]\mathbf{u}_3^T + u_4\tau_3 + u_5\tau_4 \\
E^{g}_{s+1,t} &= E^{g}_{s,t} + u_2\,E^{g\,d}_{s,t}
\end{aligned} \tag{2.54}$$

The matrices $\mathbf{I}_1$ and $\mathbf{I}_2$ are of dimension $(s+t+2\ \ \mathrm{x}\ \ s+t+3)$. They are introduced to provide symmetry to the matrix algebra and preserve initial condition calculations. We design the matrices to perform the following operations

$$[\,\omega_1 \ldots \omega_{s+1}\;\; \omega_{s+2} \ldots \omega_{s+t+2}\,]\,\mathbf{I}_1 = [\,\omega_1 \ldots \omega_{s+1}\; 0\;\; \omega_{s+2} \ldots \omega_{s+t+2}\,] \tag{2.55}$$

$$[\,\omega_1 \ldots \omega_{s+1}\;\; \omega_{s+2} \ldots \omega_{s+t+2}\,]\,\mathbf{I}_2 = [\,0\; \omega_1 \ldots \omega_{s+1}\;\; 0\; \omega_{s+2} \ldots \omega_{s+t+1}\,] \tag{2.56}$$

The values τ_1 through τ_4 can be obtained through the correlation data and forward and backward estimation parameters. We express them mathematically as

$$\begin{aligned} [\,\tau_1\;\; \tau_2\,] &= E\,[\, d_{s,t}(k-1)\,\mathbf{v}_{s,t}(k)\,] \\ \tau_3 &= -E\,[\, y(k-s-1)\, g_{s,t}(k)\,] \\ \tau_4 &= E\,[\, y(k-s-1)\, d_{s,t}(k)\,] \end{aligned} \tag{2.57}$$

The MA-type recursive formula for the forward estimation parameter matrix and backward estimation parameter vectors is obtained in a similar manner. The recursive formula is given by

$$\begin{aligned} \mathbf{C}_{s,t+1} &= \mathbf{C}_{s,t}\,\mathbf{I}_3 + \mathbf{n}_1^T\,\mathbf{G}_{s,t}\,\mathbf{I}_4 \\ \mathbf{D}_{s,t+1} &= [\,\mathbf{D}_{s,t} + n_2\,\mathbf{G}_{s,t}\,]\,\mathbf{I}_3 \\ \mathbf{G}_{s,t+1} &= \mathbf{G}_{s,t}\,\mathbf{I}_4 + [\,\mathbf{n}_3\,\mathbf{C}_{s,t} + n_4\,\mathbf{D}_{s,t} + n_5\,\mathbf{G}_{s,t}\,]\,\mathbf{I}_3 \end{aligned} \tag{2.58}$$

where

$$\begin{aligned} \mathbf{n}_1 &= -\,(E^{g}_{s,t})^{-1}\,[\,\tau'_1\;\; \tau'_2\,] \\ n_2 &= \frac{-E^{g\,d}_{s,t}}{E^{g}_{s,t}} \\ \mathbf{n}_3 &= -\,[\,\tau'_1\;\; \tau'_2\,]\,\mathbf{E}^{-1}_{s,t} \\ n_4 &= \frac{(E^{g\,d}_{s,t}\,\tau'_4 - E^{g}_{s,t}\,\tau'_3)}{\left[\,(E^{g\,d}_{s,t})^2 - E^{g}_{s,t}\,E^{d}_{s,t}\,\right]} \\ n_5 &= n_4\,n_2 \end{aligned} \tag{2.59}$$

and the (s,t + 1) prediction error powers are calculated using the following recursive formulas

$$\begin{aligned} \mathbf{E}_{s,t+1} &= \mathbf{E}_{s,t} + \mathbf{n}_1^T\,[\,\tau'_1\;\; \tau'_2\,] \\ E^{g\,d}_{s,t+1} &= \tau'_3 + n_2\,\tau'_4 \\ E^{g}_{s,t+1} &= E^{g}_{s,t} + [\,\tau'_1\;\; \tau'_2\,]\,\mathbf{n}_3^T + n_4\,\tau'3 + n_5\,\tau'_4 \\ E^{d}_{s,t+1} &= E^{d}_{s,t} + n_2\,E^{g\,d}_{s,t} \end{aligned} \tag{2.60}$$

where $\mathbf{I}_3$ and $\mathbf{I}_4$ are $(s+t+2 \times s+t+3)$ dimensional matrices which we have designed to perform the following operations

$$[\ \omega_1 \cdots \omega_{s+1}\ \ \omega_{s+2} \cdots \omega_{s+t+2}\ ]\ \mathbf{I}_3 = [\ \omega_1 \cdots \omega_{s+1}\ \ \omega_{s+2} \cdots \omega_{s+t+2}\ 0\ ] \tag{2.61}$$

$$[\ \omega_1 \cdots \omega_{s+1}\ \ \omega_{s+2} \cdots \omega_{s+t+2}\ ]\ \mathbf{I}_4 = [\ 0\ \omega_1 \cdots \omega_s\ \ 0\ \omega_{s+2} \cdots \omega_{s+t+2}\ ] \tag{2.62}$$

The values of τ'_1 through τ'_4 are calculated using correlation data in conjunction with current forward and backward parameter estimation values. We express these quantities mathematically as

$$\begin{aligned} [\ \tau'_1\ \ \tau'_2\ ] &= -E[\ g_{s,t}(k-1)\,\mathbf{v}_{s,t}(k)\ ] \\ \tau'_3 &= -E[\ x(k-t-1)\,d_{s,t}(k)\ ] \\ \tau'_4 &= E[\ x(k-t-1)\,g_{s,t}(k)\ ] \end{aligned} \tag{2.63}$$

It is interesting to note that the MA-type recursive formula is the complimentary form of the AR-type formula and that the two are identical if the variables associated with the input signal $x(k)$ and the variables associated with the ouput signal $y(k)$ are interchanged. That is, we replace $y(k)$, $\mathbf{G}_{s,t}$ and $g_{s,t}(k)$ with $-x(k)$, $\mathbf{D}_{s,t}$ and $d_{s,t}(k)$ and vice versa.

1. Experimental Results

The ARMA parameter estimation algorithm of [Ref. 4: pp. 619-621] based on the recursive formulas of equations (2.52) and (2.58) was implemented using the Fortran program found in Appendix A. This program calls subroutines which compute the ARMA model parameters as the AR order is increased by one and as the MA order is increased by one. These subroutines are shown in Appendix B and Appendix C, respectively. In the main program, an input data sequence of white Gaussian noise is passed through a known reference model producing an ouput data sequence. We obtain autocorrelation and crosscorrelation data from these input and output sequences. The correlation data is used to calculate initial values of the error powers for x and y as well as τ_1 through τ_4 and τ'_1 through τ'_4. Next we obtain estimates of the reference model by employing the recursive formulas (2.48) through (2.50), (2.52) and (2.58). Several reference models were estimated beginning with a strictly AR process of order $s = 4$ having as its transfer function

$$H(z) = \frac{1}{1 - 0.2\,z^{-1} + 0.62\,z^{-2} - 0.152\,z^{-3} + 0.3016\,z^{-4}} \tag{2.64}$$

The actual values of the reference model parameters and the ARMA model parameters which estimate this reference model are listed below

	ACTUAL	ESTIMATED
AR:	0.2000	0.2005831
	-0.6200	-0.6207655
	0.1520	0.1527565
	-0.3016	-0.3020376
MA:	1.0000	1.0001506

We next consider a second reference model with MA order $t = 2$ and AR order $s = 3$ having transfer function

$$H(z) = \frac{0.5 - 0.40\, z^{-1} + 0.89\, z^{-2}}{1 - 0.20\, z^{-1} - 0.25\, z^{-2} + 0.05\, z^{-3}} \tag{2.65}$$

The true reference model parameters and ARMA model parameter estimates are shown to be

	ACTUAL	ESTIMATED
AR:	0.2000	0.1993060
	0.2500	0.2496567
	-0.0500	-0.0491961
MA:	0.5000	0.5002602
	-0.4000	-0.3997071
	0.8900	0.8894749

A third example with MA order $t = 2$ and AR order $s = 4$ having transfer function

$$H(z) = \frac{1 + 0.2\, z^{-1} - 0.99\, z^{-2}}{1 - 0.2\, z^{-1} + 0.62\, z^{-2} - 0.152\, z^{-3} + 0.3016\, z^{-4}} \tag{2.66}$$

was considered for which we obtained the following actual and estimated reference model parameter values

	ACTUAL	ESTIMATED
AR:	0.2000	0.2011805
	-0.6200	-0.6223803
	0.1520	0.1534197
	-0.3016	-0.3036823
MA:	1.0000	0.9997638
	0.2000	0.1998342
	-0.9900	-0.9886852

We consider as a final example the reference model of AR order and MA order $s = 3$ and $t = 3$, respectively, with specific transfer function

$$H(z) = \frac{0.5 - 0.95\, z^{-1} + 1.33\, z^{-2} - 0.979\, z^{-3}}{1 + 1.69\, z^{-1} - 0.962\, z^{-2} + 0.2\, z^{-3}} \tag{2.67}$$

The actual and estimated ARMA parameters are

	ACTUAL	ESTIMATED
AR:	-1.6900	-1.6981325
	0.9620	0.9690998
	-0.2000	-0.2018440
MA:	0.5000	0.4995653
	-0.9500	-0.9553509
	1.3300	1.3346767
	-0.9790	-0.9864898

The above examples demonstrate the validity of the parameter estimation algorithm of [Ref. 4: pp. 619-621]. Many reference models were estimated using this algorithm, including pure MA processes, for which accurate estimates were obtained.

D. LATTICE STRUCTURE

In section C we developed expressions for the forward and backward prediction errors, namely, those of equations (2.31), (2.32) and (2.33). From these prediction error equations we can design elementary AR, MA and ARMA lattice structures or sections. Each elementary section satisfies the orthogonal conditions as listed in equation (2.37). From the prediction error recursive formulae, equations (2.31), (2.32) and (2.33), we construct the AR-type elementary lattice section as follows. Consider the following data set of order $(s+1,t)$ consisting of input and output data elements,

$$\mathbf{h}_{s+1,t}(k)\,\mathbf{I}_1^T = [\, y(k) \ldots y(k-s) \quad -x(k) \ldots -x(k-t+1) \quad -x(k-t)\,] \tag{2.68}$$

$$\mathbf{h}_{s+1,t}(k)\,\mathbf{I}_2^T = [\, y(k-1) \ldots y(k-s-1) \quad -x(k-1) \ldots -x(k-t) \quad 0\,] \tag{2.69}$$

where $\mathbf{I}_1^T$ and $\mathbf{I}_2^T$ are the transposes of the matrices $\mathbf{I}_1$ and $\mathbf{I}_2$ defined in equations (2.55) and (2.56). We obtain a recursive relationship between the forward prediction errors $\mathbf{v}(k)$ of order $(s+1,t)$ and order (s,t) by substituting equations (2.52), (2.68) and (2.69) in equation (2.31) such that

$$\begin{aligned} \mathbf{v}_{s+1,t}(k) &= \mathbf{h}_{s+1,t}(k)\,\mathbf{C}_{s+1,t}^T \\ &= \mathbf{h}_{s+1,t}(k)\left[\, \mathbf{I}_1^T\,\mathbf{C}_{s,t}^T + \mathbf{I}_2\,\mathbf{D}_{s,t}^T\,\mathbf{u}_1 \,\right] \\ &= \mathbf{v}_{s,t}(k) + d_{s,t}(k-1)\,\mathbf{u}_1 \end{aligned} \tag{2.70}$$

The backward prediction error recursions are obtained in a similar manner and the AR-type error recursions are

$$\begin{aligned} v_{s+1,t}^x(k) &= v_{s,t}^x(k) - u_1^x\; d_{s,t}(k-1) \\ v_{s+1,t}^y(k) &= v_{s,t}^y(k) + u_1^y\, d_{s,t}(k-1) \\ g_{s+1,t}(k) &= g_{s,t}(k) - u_2\, d_{s,t}(k) \\ d_{s+1,t}(k) &= d_{s,t}(k-1) + [\, u_3^x \;\; u_3^y \,]\, \mathbf{v}_{s,t}(k)^T - u_4\, g_{s+1,t}(k) \end{aligned} \tag{2.71}$$

where $\mathbf{u}_1 = [\, u_1^x \;\; u_1^y \,]$ and $\mathbf{u}_3 = [\, u_3^x \;\; u_3^y \,]$. The AR-type elementary lattice inverse section based on these error recursions is shown in Figure 5.

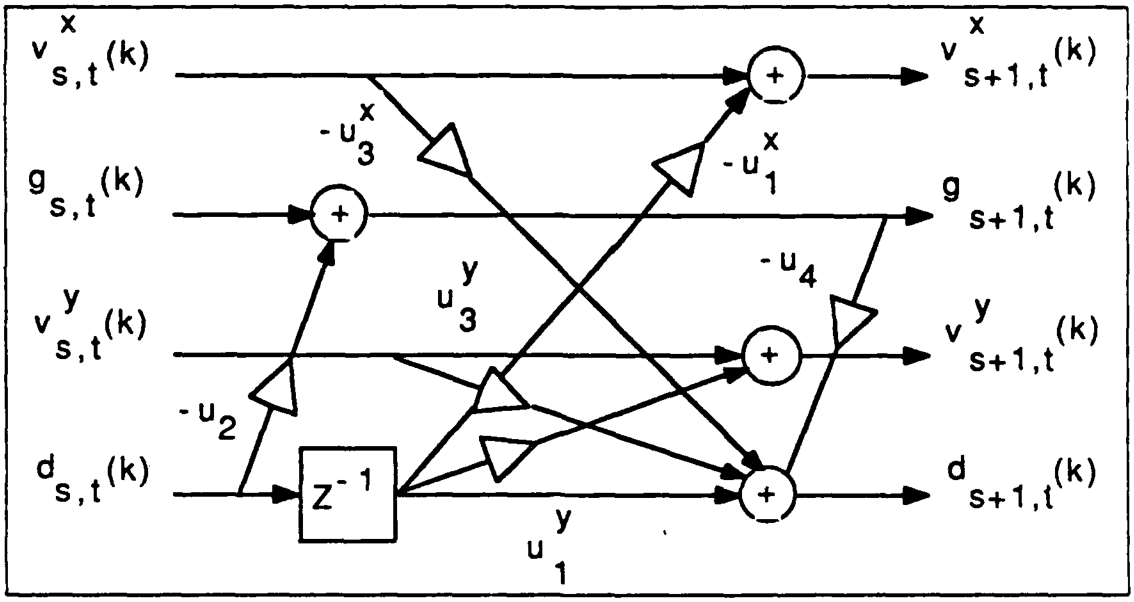

Figure 5. AR-type elementary lattice inverse section

We design the MA-type elementary lattice inverse section in a similar manner using the following representation of the data set of order $(s,t+1)$

$$\begin{aligned} \mathbf{h}_{s,t+1}(k)\,\mathbf{I}_3^T &= [\; y(k)\, y(k-1) \ldots y(k-s) \quad -x(k) \ldots -x(k-t+1) \quad -x(k-t)\;] \\ \mathbf{h}_{s,t+1}(k)\,\mathbf{I}_2^T &= [\; y(k-1) \ldots y(k-s)\;\; 0 \quad -x(k-1) \ldots -x(k-t-1)\;] \end{aligned} \tag{2.72}$$

and by substituting equations (2.58) and (2.72) into equation (2.31), we obtain the forward prediction error recursion as the MA order is increased by one, namely,

$$\begin{aligned} v^x_{s,t+1}(k) &= v^x_{s,t}(k) + u_1^x\, g_{s,t}(k-1) \\ v^y_{s,t+1}(k) &= v^y_{s,t}(k) - u_1^y\, g_{s,t}(k-1) \end{aligned} \tag{2.73}$$

The backward prediction error relationships are obtained in a similar manner and are given by

$$\begin{aligned} d_{s,t+1}(k) &= d_{s,t}(k) - u_2\, g_{s,t}(k) \\ g_{s,t+1}(k) &= g_{s,t}(k-1) - [\; u_3^x \;\; u_3^y \;]\, \mathbf{v}_{s,t}(k)^T - u_4\, d_{s,t+1}(k) \end{aligned} \tag{2.74}$$

The MA-type elementary lattice inverse section based on these error recursions is shown in Figure 6.

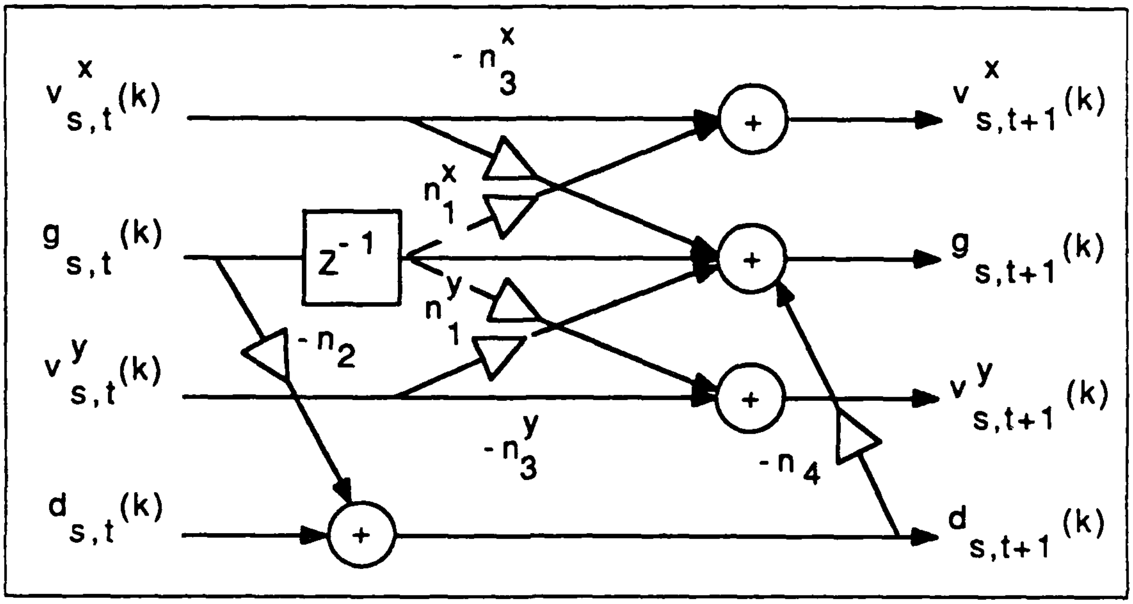

Figure 6. MA-type elementary inverse lattice section

We now construct the ARMA elementary lattice section from the AR and MA prediction error recursions. Assuming that the prediction errors are known for a given model order (s,t), the (s+1,t) prediction errors can be calculated. These prediction errors of order (s+1,t) are then updated as the MA order increases by one resulting in a prediction error of order (s+1,t+1). We consider the forward prediction error for x as the AR order is increased by one, specifically

$$v^x_{s+1,t}(k) = v^x_{s,t}(k) - u^x_1 d_{s,t}(k-1) \tag{2.75}$$

Now, $v^x_{s-1,t}(k)$ becomes the current value of the forward prediction error for x and when we calculate the (s+1,t+1) forward prediction error we have from equation (2.73)

$$v^x_{s+1,t+1}(k) = v^x_{s+1,t}(k) + n^x_1 g_{s+1,t}(k-1) \tag{2.76}$$

Equation (2.76) can be expressed in terms of the (s,t) forward prediction errors of x by making appropriate substitutions for $v^x_{s-1,t}(k)$ and $g_{s-1,t}(k-1)$. That is, we substitute $v^x_{s-1,t}(k)$ and $g_{s-1,t}(k-1)$ of equation (2.71) in equation (2.76) to obtain the (s+1,t+1) forward prediction errors for x, namely,

$$v^x_{s+1,t+1}(k) = v^x_{s,t}(k) - u^x_1 d_{s,t}(k-1) + n^x_1[\, g_{s,t}(k-1) - u_2 d_{s,t}(k-1)\,] \tag{2.77}$$

Grouping the terms we obtain

$$v^x_{s+1,t+1}(k) = v^x_{s,t}(k) - (u^x_1 + n^x_1 u_2)\, d_{s,t}(k-1) + n^x_1\, g_{s,t}(k-1) \tag{2.78}$$

The forward prediction error recursion for y of order (s+1,t+1) is obtained in a similar manner. We begin with the (s+1,t) order update of the prediction error and after it is computed, update the MA order. Specifically, we have

$$v^y_{s+1,t}(k) = v^y_{s,t}(k) + u^y_1\, d_{s,t}(k-1) \tag{2.79}$$

and from equation (2.73)

$$v^y_{s+1,t+1}(k) = v^y_{s+1,t}(k) - n^y_1\, g_{s+1,t}(k-1) \tag{2.80}$$

Substituting (2.71) for $v^y_{s-1,t}(k)$ and $g_{s-1,t}(k-1)$ in equation (2.80) then grouping terms we obtain the (s+1,t+1) forward prediction error recursion for y,

$$v^y_{s+1,t+1}(k) = v^y_{s,t}(k) + (u^y_1 + n^y_1 u_2)\, d_{s,t}(k-1) - n^y_1\, g_{s,t}(k-1) \tag{2.81}$$

The (s+1,t+1) backward prediction errors for x and y are derived in a similar manner and are given by,

$$\begin{aligned} g_{s+1,t+1}(k) &= g_{s,t}(k-1) + (n^x_3 + n_4 u^x_3)\, v^x_{s,t}(k) - (n^y_3 + n_4 u^y_3)\, v^y_{s,t}(k) \\ d_{s+1,t+1}(k) &= d_{s,t}(k-1) - u^x_3\, v^x_{s,t}(k) + u^y_3\, v^y_{s,t}(k) \end{aligned} \tag{2.82}$$

The ARMA elementary lattice inverse section is shown in Figure 7 where the coefficients are related to the prediction error recursions by the following

$$w^1_1 = (u^x_1 + n^x_1 u_2), \quad w^1_2 = n^x_1, \quad w^1_3 = (u^y_1 + n^y_1 u_2), \quad w^1_4 = n^y_1 \tag{2.83}$$

$$w^1_5 = (n^x_3 + n_4 u^x_3), \quad w^1_6 = (n^y_3 + n_4 u^y_3), \quad w^1_7 = u^x_3, \quad w^1_8 = u^y_3 \tag{2.84}$$

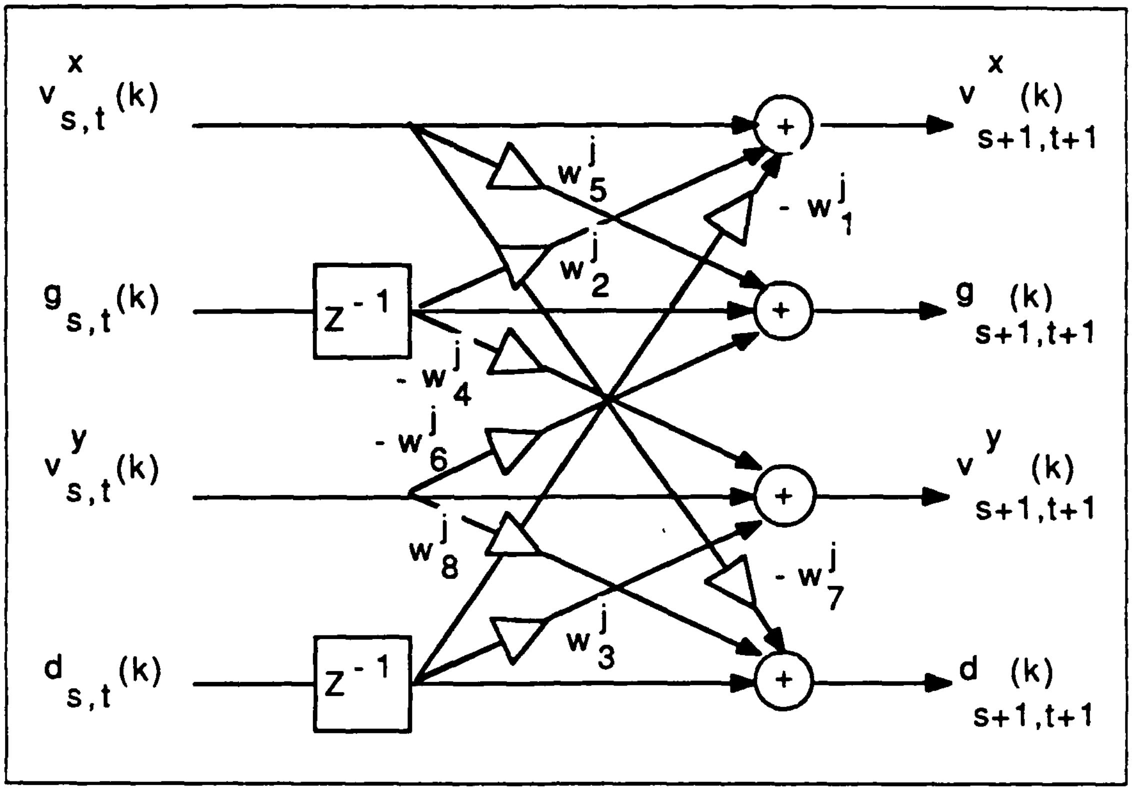

Figure 7. ARMA elementary lattice section

We see from Figure 7 that each elementary ARMA lattice inverse section contains eight coefficients.

From the AR, MA and ARMA elementary lattice inverse sections, we can obtain synthesis lattice structures. These structures provide a means of working with lattice realizations as linear filters. The resulting AR, MA and ARMA elementary synthesis lattice filters are shown in Figures 8 and 9 respectively.

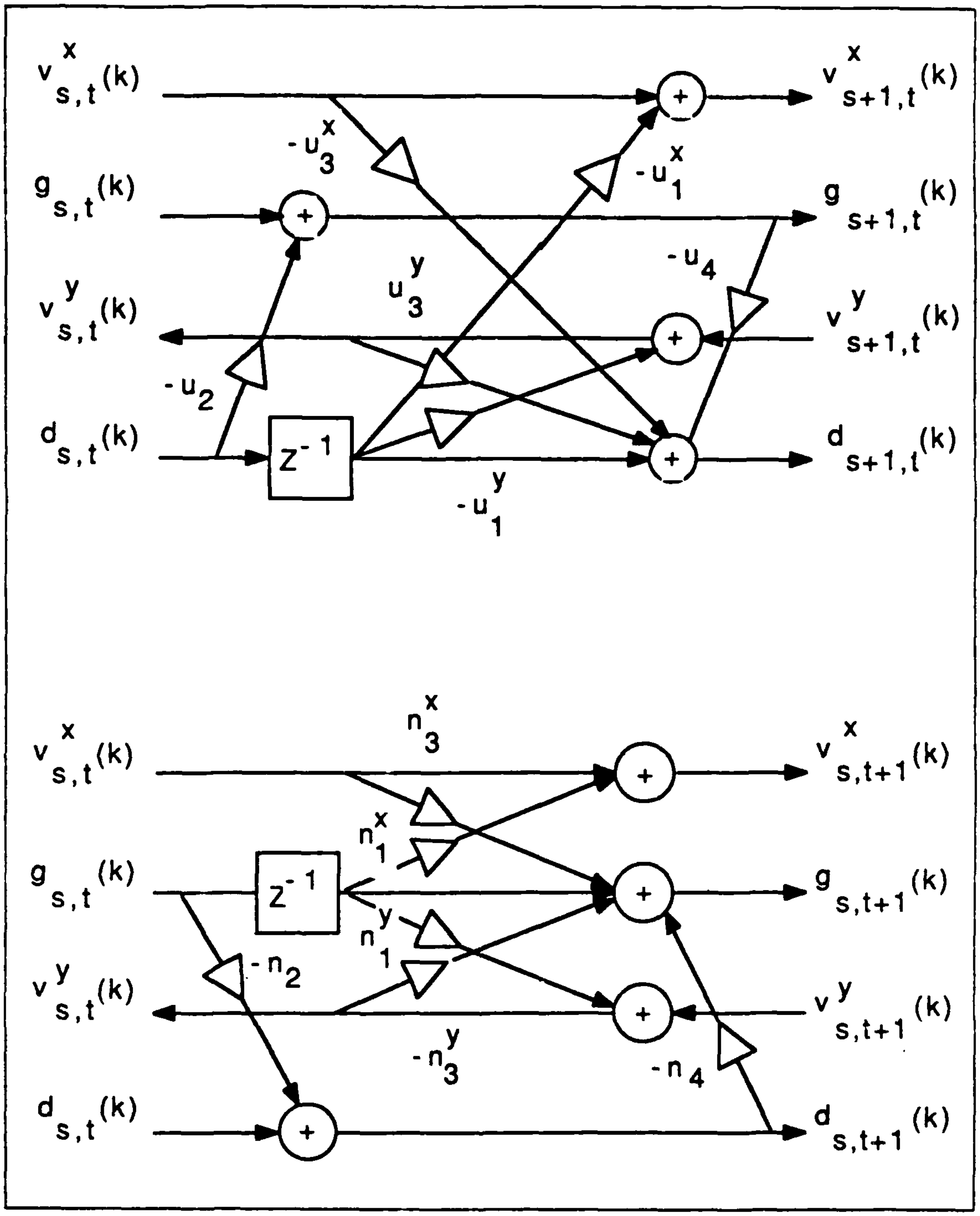

Figure 8.. Top: AR elementary lattice filter. Bottom: MA elementary lattice section.

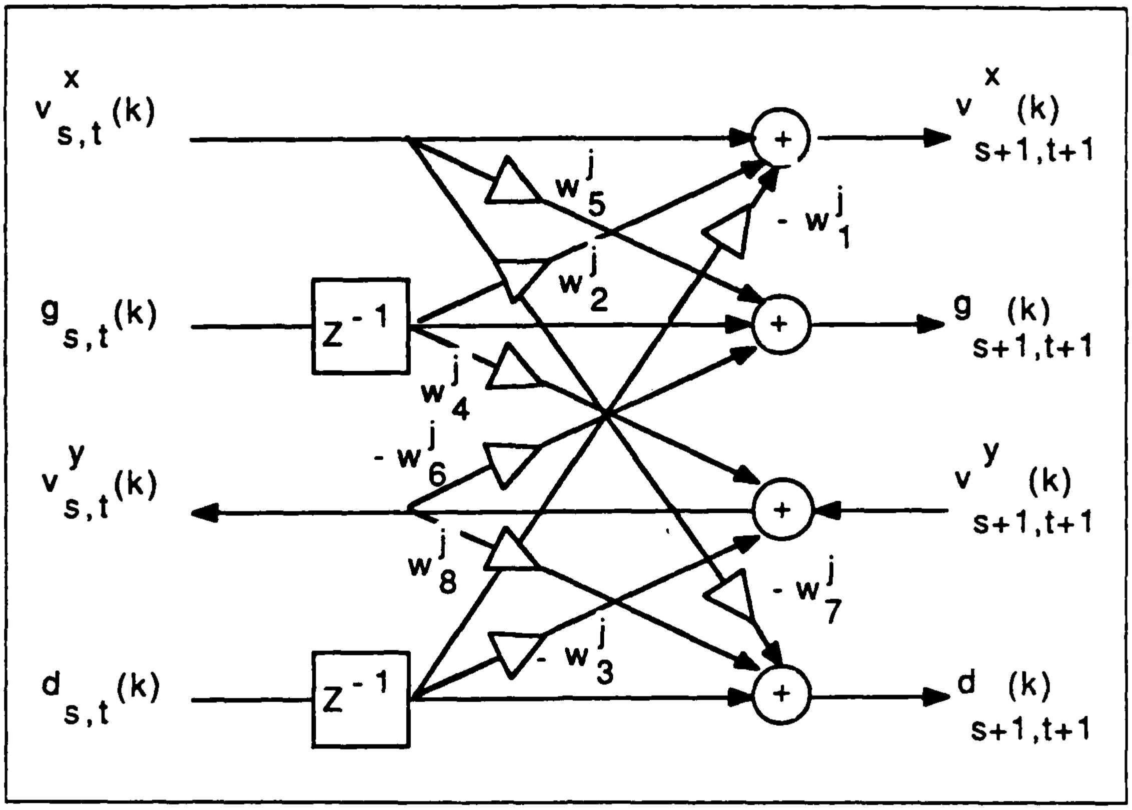

Figure 9. ARMA elementary lattice filter

Summarizing, in this chapter we have reviewed the Mullis-Roberts criterion, introduced the ARMA parameter estimation as a generalized Mullis-Roberts criterion and obtained analysis and synthesis forms of lattice structures. We notice that each ARMA elementary lattice section consists of eight reflection parameters and the calculation of these parameters requires the autocorrelation and crosscorrelation information as obtained from the input output data of the reference model. Also, we obtained a set of equations relating the final model estimation parameters and the prediction error model parameters which inturn are obtained from the eight lattice parameters.

III. ADAPTIVE LATTICE ALGORITHM

A. LEAST MEAN SQUARE ALGORITHM

The study and design of adaptive filters is known to be a very important part of statistical signal processing. Many adaptive algorithms have been developed to support the application of adaptive filtering in communications and control [Ref. 12]. An adaptive filter is characterized by the ability of its filter coefficients to adjust (self-optimize) automatically and yield an optimum filter design. Two processes occur within an adaptive filter, namely, the adaptation and the filtering processes. During the filtering process a desired signal is applied to an adaptive algorithm as a reference for adjusting the filter coefficients. Figure 10 shows a block diagram of the adaptive modeling process. Referring to Figure 10, let $y(k)$ be the output of the filter at time k. By comparing the output with the desired signal $d(k)$, an error signal $e(k)$ is generated. The adaptive algorithm of the filter uses this error signal to generate corrections which are applied to the filter coefficients such that an optimum solution is obtained. An optimization technique called the method of steepest descent provides an approach to solving this problem. The procedure is as follows:

1. Assign initial values to all filter coefficients.
2. Using these initial values, compute the gradient vector, whose individual elements equal the first derivatives of the mean-squared error with respect to the filter coefficients.
3. Compute values for the filter coefficients by changing the initial values in the direction opposite that of the gradient.
4. Return to step 2 and repeat the procedure.

There is, however, a limitation to this procedure. The steepest descent algorithm requires exact measurements of the gradient vector at each iteration which, in practice, is not possible. Therefore, the gradient vector must be estimated and consequently, errors are introduced. An algorithm is required which computes the gradient from the available data. The least mean square (LMS) algorithm, developed by Widrow and Hoff, is widely used and is very convenient to implement in real time hardware [Ref. 13: pp. 96-104]. Let $y(k)$ be the output of the filter and $d(k)$ the desired signal at time k as shown in Figure 10. We compute the error by taking the difference between these two signals, namely,

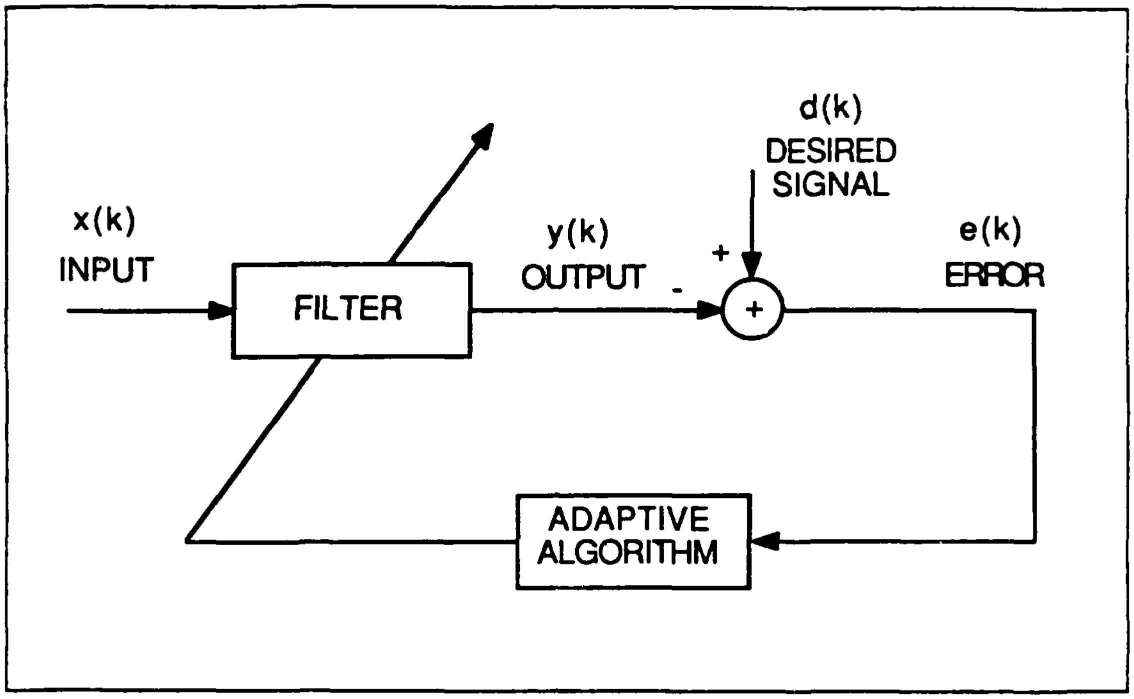

Figure 10. Adaptive modeling block diagram

$$e(k) = d(k) - y(k) \tag{3.1}$$

The value of the mean-squared error is the expected value of the error squared, $E[\, e^2(k)\,]$ and the gradient vector, $\nabla(k)$, is the first derivative of the mean-squared error. The gradient vector is given by

$$\nabla(k) = \frac{\partial}{\partial w(k)} E\left[\, e^2(k)\, \right] = 2\, e(k)\, \frac{\partial}{\partial w(k)}\, e(k) \tag{3.2}$$

where $w(k)$ is the time dependent filter coefficient vector. The recursion for the filter coefficient which changes the old value in the direction opposite to that of the gradient is then given by,

$$\begin{aligned} w(k) &= w(k-1) + \frac{1}{2}\, \mu\, [\, -\nabla(k)\,] \\ &= w(k-1) - \mu\, e(k)\, \frac{\partial}{\partial w(k)}\, e(k) \end{aligned} \tag{3.3}$$

where $w(k)$ is the filter coefficient vector estimate at the k^{th} iteration, $w(k-1)$ is the past filter coefficient vector estimate, μ is the convergence (gain) constant, $e(k)$ is the error signal at the k^{th} iteration and $\frac{\partial}{\partial \mathbf{w}(k)} e(k)$ is the instantaneous gradient. The implementation of this algorithm proceeds as follows:

1. Assign initial values to the filter coefficients.
2. Compute the value for the error signal $e(k)$.
3. Calculate the updated estimate of the filter coefficients using the instantaneous gradient.
4. Increment the time index by one and return to step 1.

Convergence properties of the LMS algorithm are well documented within the literature. The choice of a gain constant μ is arbitrary however, theoretical bounds have been derived for μ, given by [Ref. 14: pp. 101-106],

$$0 < \mu < \frac{2}{\lambda_{\max}} \simeq \frac{1}{\mathrm{Tr}[\mathbf{R}_{xx}]} \tag{3.4}$$

where $\lambda_{\max}$ is the maximum eigenvalue of the input autocorrelation matrix, $\mathbf{R}_{xx}$, and where Tr [$\mathbf{R}_{xx}$] is the trace of the matrix $\mathbf{R}_{xx}$.

B. DERIVATION OF THE ADAPTIVE LATTICE ALGORITHM

The adaptive lattice algorithm developed in this thesis uses concepts of the LMS algorithm discussed in section A and applies them to the ARMA digital lattice filter proposed in Chapter II. Consider the ARMA digital lattice filter of Figure 11, which consists of two cascaded elementary lattice sections. The filter coefficients (weights) are defined such that w_i^m represents the i^{th} lattice coefficient at stage m of the lattice structure. In this figure we have a two stage lattice and there are eight coefficients per elementary lattice section. The output, $\hat{y}(k)$, of the lattice filter can be determined from

$$\hat{y}(k) = e_{f_1}^{y}(k) + w_4^1\, e_{b_0}^{x}(k-1) - w_1^1\, e_{b_0}^{y}(k-1) \tag{3.5}$$

Forward errors at a given stage m of the lattice filter are defined as,

$$\begin{aligned} e_{f_m}^{x}(k) &= e_{f_{m-1}}^{x}(k) + w_2^m\, e_{b_{m-1}}^{x}(k-1) - w_1^m\, e_{b_{m-1}}^{y}(k-1) \\ e_{f_m}^{y}(k) &= e_{f_{m-1}}^{y}(k) + w_4^{m+1}\, e_{b_m}^{x}(k-1) - w_3^{m+1}\, e_{b_m}^{y}(k-1) \end{aligned} \tag{3.6}$$

and the backward errors for any given stage m are,

$$\begin{aligned} e_{b_m}^{x}(k) &= e_{b_{m-1}}^{x}(k-1) + w_5^m\, e_{f_{m-1}}^{x}(k) - w_6^m\, e_{f_{m-1}}^{y}(k) \\ e_{b_m}^{y}(k) &= e_{b_{m-1}}^{y}(k-1) - w_7^m\, e_{f_{m-1}}(k) + w_8^m\, e_{f_{m-1}}^{y}(k) \end{aligned} \tag{3.7}$$

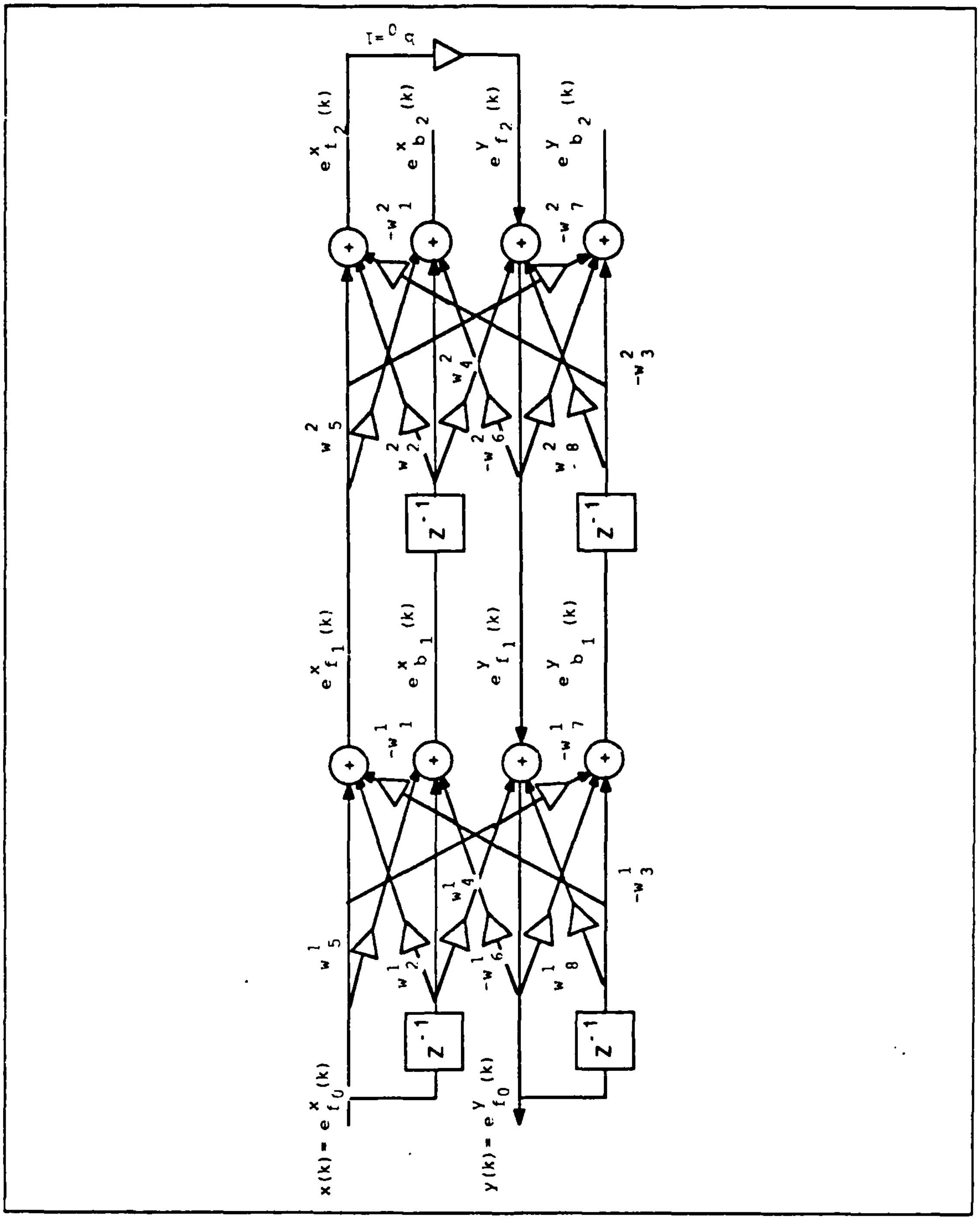

Figure 11. Two stage ARMA lattice digital filter.

where, in Figure 11, $m = 1,2$ and $e^x_{f_0} = x(k)$ and $e^y_{f_0} = \hat{y}(k)$. The terminal condition is $e^y_{f_m}(k) = b_0\, e^x_{f_m}(k)$; $m = 2$. To begin with, let b_0 equal unity. The initial conditions are $e^y_{b_0}(k-1) = 0$ and $e^x_{b_0}(k-1) = 0$. As with the LMS algorithm, we form an error between a desired signal $d(k)$ and the output signal $\hat{y}(k)$ such that,

$$e(k) = d(k) - \hat{y}(k) \tag{3.8}$$

The instantaneous gradient according to eq (3.2) is then,

$$\nabla(k) = 2\, e(k)\, \frac{\partial}{\partial \mathbf{w}(k)} \left[\, d(k) - \hat{y}(k)\, \right] \tag{3.9}$$

Since the desired signal, $d(k)$, is not a function of the filter coefficients, equation (3.9) reduces to,

$$\nabla(k) = 2\, e(k)\, \frac{\partial}{\partial \mathbf{w}(k)} \left[\, -\hat{y}(k)\, \right] \tag{3.10}$$

where the quantity $\frac{\partial}{\partial \mathbf{w}(k)} \left[\, -\hat{y}(k)\, \right]$ is referred to as the gradient estimator. This gradient estimator must be computed for each filter coefficient within the lattice structure. The filter coefficients are then updated using the respective gradient estimators. That is, we need to compute,

$$\nabla(k) = \frac{\partial}{\partial\, w_i^j}\, \hat{y}(k) \;\; \text{for } i = 1,2,\ldots,8 \text{ and } j = 1,2,\ldots,M \tag{3.11}$$

where M is the number of stages in the ARMA lattice filter. From equation (3.5), the gradient estimates are given by

$$\begin{aligned} \frac{\partial \hat{y}(k)}{\partial w_i^j} = {} & \frac{\partial e^y_{f_1}(k)}{\partial w_i^j} + \frac{\partial w_4^1}{\partial w_i^j}\, e^x_{b_0}(k-1) + w_4^1\, \frac{\partial e^x_{b_0}(k-1)}{\partial w_i^j} - \frac{\partial w_3^1}{\partial w_i^j}\, e^y_{b_0}(k-1) \\ & - w_3^1\, \frac{\partial e^y_{b_0}(k-1)}{\partial w_i^j} \end{aligned} \tag{3.12}$$

Let $\psi(k)$ represent the partial derivatives of the output $\hat{y}(k)$ with respect to the filter coefficients and $\phi(k)$ represent the partial derivatives of the errors with respect to the filter coefficients. Using this representation we can re-write equation (3.12) as follows,

$$\psi_{ij}(k) = \phi^{y}_{f_1\,ij}(k) + \delta^{1j}_{4i}\, e^{x}_{b_0}(k-1) + w^{1}_{4}\, \phi^{x}_{b_0\,ij}(k-1) - \delta^{1j}_{3i}\, e^{v}_{b_0}(k-1) - w^{1}_{3}\, \phi^{v}_{b_0\,ij}(k-1) \tag{3.13}$$

where δ^{1j}_{4i}, δ^{1j}_{3i}, are kronecker deltas whose value is one if and only if $i=4$ and $j=1$ or $i=3$ and $j=1$ respectively. We compute $\psi_{ij}(k)$, by obtaining recursive relations which calculate the partial derivative of the forward and backward errors with respect to the filter coefficients. These relations are obtained from equations (3.6) and (3.7) by taking the partial derivatives with respect to the filter coefficients, namely,

$$\phi^{x}_{f_m\,ij}(k) = \phi^{x}_{f_{m-1}\,ij}(k) + \delta^{mj}_{2i}\, e^{x}_{b_{m-1}}(k-1) + w^{m}_{2}\, \phi_{b_{m-1}\,ij}{}^{x}(k-1) - \delta^{mj}_{1i}\, e^{v}_{b_{m-1}}(k-1) - w^{m}_{1}\, \phi^{v}_{b_{m-1}\,ij}(k-1)$$

$$\phi^{y}_{f_m\,ij}(k) = \phi^{y}_{f_{m-1}\,ij}(k) + \delta^{(m+1)j}_{4i}\, e^{x}_{b_m}(k-1) + w^{m+1}_{4}\, \phi^{x}_{b_m\,ij}(k-1) - \delta^{(m+1)j}_{3i}\, e^{v}_{b_m}(k-1) - w^{m+1}_{3}\, \phi^{v}_{b_m\,ij}(k-1)$$

$$\phi^{x}_{b_m\,ij}(k) = \phi^{x}_{b_{m-1}\,ij}(k-1) + \delta^{mj}_{5i}\, e^{x}_{f_{m-1}}(k) + w^{m}_{5}\, \phi^{x}_{f_{m-1}\,ij}(k) - \delta^{mj}_{6i}\, e^{y}_{f_{m-1}}(k) - w^{m}_{6}\, \phi^{v}_{f_{m-1}\,ij}(k) \tag{3.14}$$

$$\phi^{v}_{b_m\,ij}(k) = \phi^{v}_{b_{m-1}\,ij}(k-1) - \delta^{mj}_{7i}\, e^{x}_{f_{m-1}}(k) - w^{m}_{7}\, \phi^{x}_{f_{m-1}\,ij}(k) + \delta^{mj}_{8i}\, e^{y}_{f_{m-1}}(k) + w^{m}_{8}\, \phi^{v}_{f_{m-1}\,ij}(k)$$

These recursive relations possess a lattice structure similar to that of Figure 11, with delta components injected at the summation nodes. They may also be simplified by examining individual terms. Consider the general equation for the forward error in x, repeated here for continuity.

$$e^{x}_{f_m}(k) = e^{x}_{f_{m-1}}(k) + w^{m}_{2}\, e^{x}_{b_{m-1}}(k-1) - w^{m}_{1}\, e^{v}_{b_{m-1}}(k-1) \tag{3.15}$$

The partial derivative with respect to each filter coefficient is expressed as,

$$\frac{\partial e^{x}_{f_m}(k)}{\partial w^{j}_{i}} = \frac{\partial e^{x}_{f_{m-1}}(k)}{\partial w^{j}_{i}} + \frac{\partial w^{m}_{2}}{\partial w^{j}_{i}}\, e^{x}_{b_{m-1}}(k-1) + w^{m}_{2}\, \frac{\partial e^{x}_{b_{m-1}}(k-1)}{\partial w^{j}_{i}} - \frac{\partial w^{m}_{1}}{\partial w^{j}_{i}}\, e^{v}_{b_{m-1}}(k-1) - w^{m}_{1}\, \frac{\partial e^{v}_{b_{m-1}}(k-1)}{\partial w^{j}_{i}} \tag{3.16}$$

Since the partial derivative is taken with respect to the current filter coefficient w_i^j at time k, the partial derivatives involving delay terms i.e., $(k-1)$, are set to zero. This result follows from the realistic assumption that $e_{b_m}^x(k-1)$ is a function of $w_i^j(k-1)$ but not of $w_i^j(k)$. Also note that $w_i^j(k)$ is a function of $w_i^j(k-1)$ but not vice versa. With these simplifications we reduce the equations of (3.14) to,

$$
\begin{aligned}
\phi_{f_m ij}^x(k) &= \phi_{f_{m-1} ij}^x(k) + \delta_{2\,i}^{mj}\, e_{b_{m-1}}^x(k-1) - \delta_{1\,i}^{mj}\, e_{b_{m-1}}^y(k-1) \\
\phi_{f_m ij}^y(k) &= \phi_{f_{m-1} ij}^y(k) + \delta_{4\,i}^{(m+1)j}\, e_{b_m}^x(k-1) - \delta_{3\,i}^{(m+1)j}\, e_{b_m}^y(k-1) \\
\phi_{b_m ij}^x(k) &= \delta_{5\,i}^{mj}\, e_{f_{m-1}}^x(k) + w_5^m\, \phi_{f_{m-1} ij}^x(k) - \delta_{6\,i}^{mj}\, e_{f_{m-1}}^y(k) - w_6^m\, \phi_{f_{m-1} ij}^y(k) \\
\phi_{b_m ij}^y(k) &= -\,\delta_{7\,i}^{mj}\, e_{f_{m-1}}^x(k) - w_7^m\, \phi_{f_{m-1} ij}^x(k) + \delta_{8\,i}^{mj}\, e_{f_{m-1}}^y(k) + w_8^m\, \phi_{f_{m-1} ij}^y(k)
\end{aligned}
\tag{3.17}
$$

and the gradient estimator is,

$$
\psi_{ij}(k) = \phi_{f_1 ij}^y(k) + \delta_{4\,i}^{1j}\, e_{b_0}^x(k-1) - \delta_{3\,i}^{1j}\, e_{b_0}^y(k-1) \tag{3.18}
$$

Although these are valid recursive relations, they are difficult to implement in a lattice algorithm. The ultimate goal is the requirement to easily compute $\psi_{ij}(k)$ from the available data. From eq (3.18) it is evident that $\psi_{ij}(k)$ depends on $\phi_{f_1 ij}^y(k)$ which in turn requires knowledge of $\phi_{f_m ij}^y(k)$ and $\phi_{f_m ij}^x(k)$ but not of $\phi_{b_m ij}^y(k)$ or $\phi_{b_m ij}^x(k)$. Therefore the three equations necessary to compute the gradient estimator are,

$$
\begin{aligned}
\psi_{ij}(k) &= \phi_{f_1 ij}^y(k) + \delta_{4\,i}^{1j}\, e_{b_0}^x(k-1) - \delta_{3\,i}^{1j}\, e_{b_0}^y(k-1) \\
\phi_{f_m ij}^x(k) &= \phi_{f_{m-1} ij}^x(k) + \delta_{2\,i}^{mj}\, e_{b_{m-1}}^x(k-1) - \delta_{1\,i}^{mj}\, e_{b_{m-1}}^y(k-1) \\
\phi_{f_m ij}^y(k) &= \phi_{f_{m-1} ij}^y(k) + \delta_{4\,i}^{(m+1)j}\, e_{b_m}^x(k-1) - \delta_{3\,i}^{(m+1)j}\, e_{b_m}^y(k-1)
\end{aligned}
\tag{3.19}
$$

These equations are dependent on the filter coefficients w_i^j, $i = 1,2,3,4$ and $j = 1,2,\ldots,M$, thereby reducing by one-half the number of computations required for $\phi_{f_m ij}^x(k)$ and $\phi_{f_m ij}^y(k)$. A recursive relation is desired for $\psi_{ij}(k)$ which does not involve delta functions. Consider the four stage lattice filter with terminal condition b_0 equal to unity such that $\phi_{f_M ij}^y(k) = \phi_{f_M ij}^x(k)$. The procedure for computing $\psi_{ij}(k)$ using the equations of (3.19) is as follows:

1. Calculate $\phi_{f_m ij}^x(k)$ with m equal to one and letting i and j range from one to four. Repeat for $m = 2,3,4$.
2. Using the terminal condition and expression for $\phi_{f_m ij}^y(k)$, calculate $\phi_{f_1 ij}^y(k)$.

This requires solving 88 equations, however, the result is a very simple recursive formula for $\psi_{ij}(k)$, namely,

$$\psi_{ij}(k) = -\, e^{v}_{b_{j-1}}(k-1) \qquad i = 1\,,\,3$$
$$\psi_{ij}(k) = e^{x}_{b_{j-1}}(k-1) \qquad i = 2\,,\,4 \tag{3.20}$$

The lattice coefficients are calculated by substituting the recursive formula of eq (3.20) for the gradient estimator in equation (3.3). The adaptive coefficient update equation is,

$$w_i^j(k) = w_i^j(k-1) - \mu\, e(k)\, \psi_{ij}(k) \tag{3.21}$$

Since the gradient estimator, and therefore the gradient, is the same for w_i^j , $i = 1,3$ and w_i^j, $i = 2,4$, it follows that $w_1^j = w_3^j$ and $w_2^j = w_4^j$. The number of filter coefficients required to update the lattice filter is reduced to two, i.e., w_1^j and w_2^j. Furthermore, from the symmetry of the lattice structure, the following equalities between filter coefficients are assumed,

$$\begin{aligned} w_2^j &= w_5^j \\ w_1^j &= w_7^j \\ w_6^j &= w_4^j \\ w_3^j &= w_8^j \end{aligned} \tag{3.22}$$

Incorporating these equalities with those derived by the gradient estimator produces the elementary ARMA lattice section of Figure 12 where,

$$\begin{aligned} w_2^j &= w_4^j = w_5^j = w_6^j = r_j \\ w_1^j &= w_3^j = w_7^j = w_8^j = k_j \end{aligned} \tag{3.23}$$

To prove that these coefficient reductions are valid, a computer generated solution using the Fortran program of Appendix D was compared to hand analysis of a second order transfer function and lattice filter. The output of the ARMA digital lattice filter was first put into difference equation form and then compared to the known transfer function. From this comparison, lattice coefficients were computed. Details of this analysis are as follows. Consider a two stage ARMA digital lattice filter comprised of the reduced elementary section shown in Figure 13 and a transfer function of the form,

$$H_f(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \tag{3.24}$$

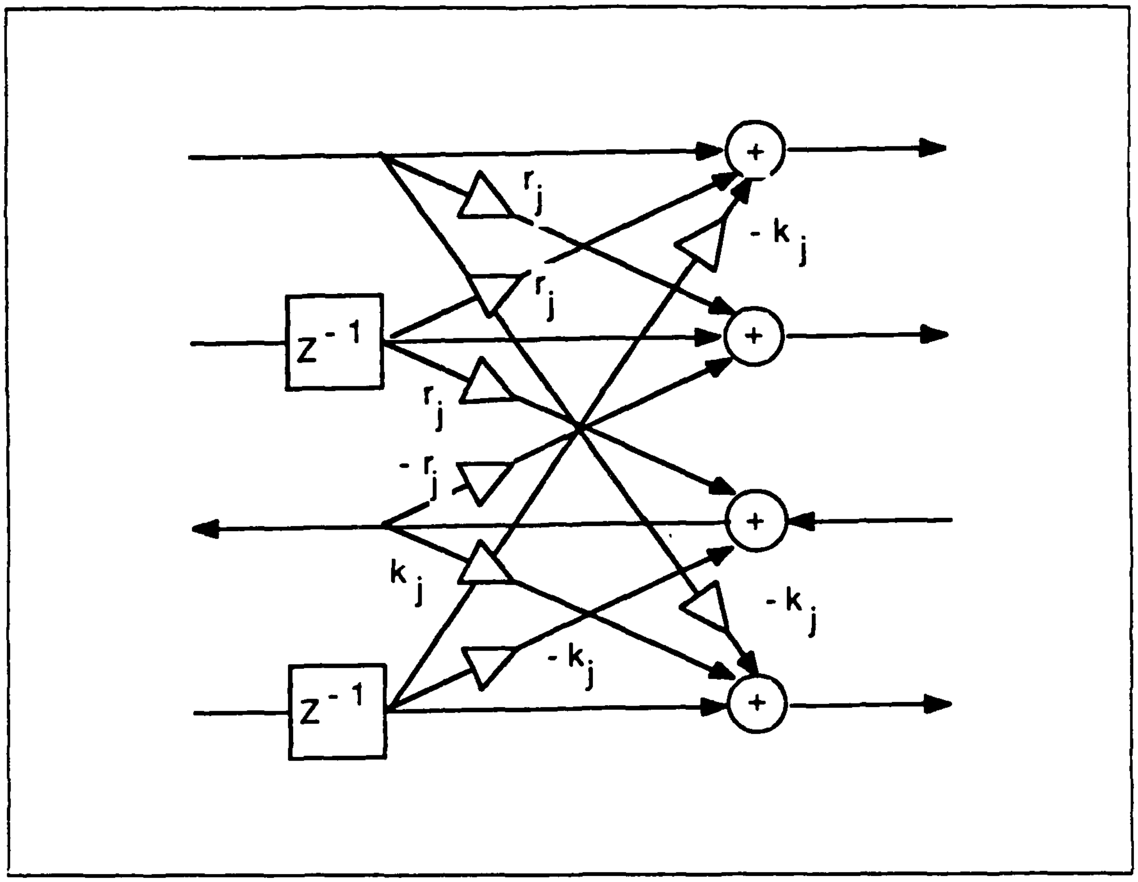

Figure 12. Simplified elementary ARMA lattice section.

The output of the lattice filter can be written in difference equation form by carrying out the following steps: (i) start with the output of the lattice filter equation (3.5), (ii) substitute expressions for the forward and backward errors, equations (3.6) and (3.7), respectively, into equation (3.5) and (iii) carry out the algebra. A detailed derivation of this difference equation is given in Appendix E. The difference equation in its final form is given by,

$$\begin{aligned} y(k) = x(k) &+ 2(w_2^1 + w_1^1 w_1^2 + w_2^1 w_2^2)x(k-1) + 2 w_2^2 x(k-2) \\ &-2(w_1^1 + w_2^1 w_2^2 + w_1^1 w_1^2)y(k-1) - 2 w_1^2 y(k-2) \end{aligned} \tag{3.25}$$

which can be written in the transfer function form as,

$$H_l(z) = \frac{1 + 2(\, w_2^1 \,+\, w_1^1\, w_1^2 \,+\, w_2^1\, w_2^2\,)\, z^{-1} + 2\, w_2^2\, z^{-2}}{1 + 2(\, w_1^1 \,+\, w_2^1\, w_2^2 \,+\, w_1^1\, w_1^2\,)\, z^{-1} + 2\, w_1^2\, z^{-2}} \tag{3.26}$$

Comparing the lattice filter transfer function, $H_l(z)$ with the known filter transfer function $H_f(z)$, produces the following relationships between filter coefficients,

$$\begin{aligned} b_1 &= 2(\, w_2^1 \,+\, w_1^1\, w_1^2 \,+\, w_2^1\, w_2^2\,) \\ a_1 &= 2(\, w_1^1 \,+\, w_2^1\, w_2^2 \,+\, w_1^1\, w_1^2\,) \\ b_2 &= 2\, w_2^2 \\ a_2 &= 2\, w_1^2 \end{aligned} \tag{3.27}$$

Solving for w_2^2 and w_1^2 in terms of the known transfer function coefficients b_2 and a_2 and then substituting these results into the expressions for b_1 and a_1, respectively, yields the following.

$$\begin{aligned} b_1 &= a_2\, w_1^1 \,+\, (\,2 + b_2\,)\, w_2^1 \\ a_1 &= (\,2 + a_2\,)\, w_1^1 \,+\, b_2\, w_2^1 \end{aligned} \tag{3.28}$$

or in matrix form,

$$\begin{bmatrix} b_1 \\ a_1 \end{bmatrix} = \begin{bmatrix} a_2 & 2 + b_2 \\ 2 + a_2 & b_2 \end{bmatrix} \begin{bmatrix} w_1^1 \\ w_2^1 \end{bmatrix} \tag{3.29}$$

and solving for the lattice coefficients, we have

$$\begin{bmatrix} w_1^1 \\ w_2^1 \end{bmatrix} = \frac{1}{a_2\, b_2 \,-\, (2 + b_2)\,(2 + a_2)} \begin{bmatrix} b_2 & -\,(2 + b_2) \\ -\,(2 + a_2) & a_2 \end{bmatrix} \begin{bmatrix} b_1 \\ a_1 \end{bmatrix} \tag{3.30}$$

and

$$\begin{aligned} w_2^2 &= \frac{b_2}{2} \\ w_1^2 &= \frac{a_2}{2} \end{aligned} \tag{3.31}$$

Now that a method of converting between lattice and transfer function coefficients for a second order system has been established, we consider the specific transfer function

$$H_f(z) = \frac{1 - 0.8\,z^{-1} + 1.78\,z^{-2}}{1 - 0.89\,z^{-1} + 0.25\,z^{-2}} \tag{3.32}$$

where

$$b_0 = 1.0 \quad b_1 = -0.80 \quad b_2 = 1.78$$
$$a_1 = -0.89 \quad a_2 = 0.25$$

From equations (3.30) and (3.31) the lattice coefficients are calculated as,

$$w_1^2 = 0.125, \quad w_2^2 = 0.890, \quad w_1^1 = -0.240719, \quad w_2^1 = -0.195719$$

Values for the steady-state lattice coefficients were computed using the Fortran program in Appendix D and are shown below. Convergence aspects of both the lattice coefficients and output error are shown in Figure 13.

$$w_1^2 = 0.124982, \quad w_2^2 = 0.890003, \quad w_1^1 = -0.240710, \quad w_2^1 = -0.195711$$

these results confirm the validity of the derived adaptive lattice algorithm and the design of a new elementary lattice section shown in Figure 12.

The current adaptive lattice algorithm assumes that the terminal condition is unity. This is generally not the case in practice. We now extend this adaptive algorithm to the more general case where the terminal condition is an arbitrary constant. The recursive relation which updates b_0 is similar to those which update the other lattice filter coefficients. The update equation for b_0 is given by

$$b_0(k) = b_0(k-1) - \mu\, e(k)\, \frac{\partial e(k)}{\partial b_0(k)} \tag{3.33}$$

The gradient estimator $\frac{\partial e(k)}{\partial b_0(k)}$, is calculated using equations (3.5), (3.8) and the fact that the desired signal $d(k)$ is not dependent on b_0. The gradient estimator for b_0 is written as,

$$\frac{\partial \hat{y}(k)}{\partial b_0} = \frac{\partial e_{f_1}^{y}(k)}{\partial b_0} + \frac{\partial w_4^1}{\partial b_0}\, e_{b_0}^{x}(k-1) + w_4^1\, \frac{\partial e_{b_0}^{x}(k-1)}{\partial b_0} - \frac{\partial w_3^1}{\partial b_0}\, e_{b_0}^{y}(k-1) - w_3^1\, \frac{\partial e_{b_0}^{y}(k-1)}{\partial b_0} \tag{3.34}$$

Since the partial derivative is taken with respect to b_0 at time k, this reduces to,

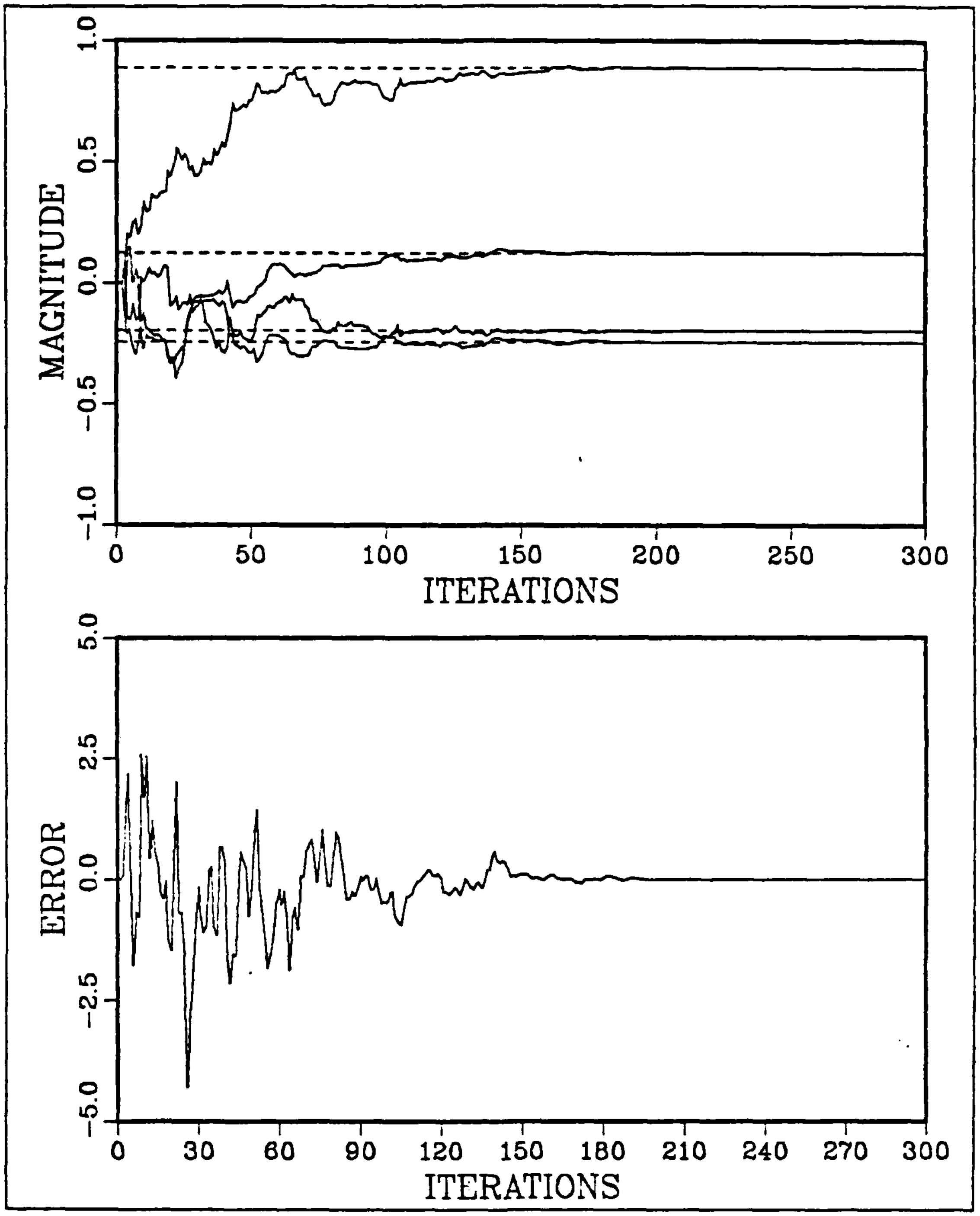

Figure 13. Top: Lattice coefficients. Bottom: Output error.

$$\frac{\partial \hat{y}(k)}{\partial b_0} = \frac{\partial e^y_{f_1}(k)}{\partial b_0} \tag{3.35}$$

Similarly,

$$e^y_{f_m}(k) = e^y_{f_{m+1}}(k) + w_4^{m+1}\, e^x_{b_m}(k-1) - w_3^{m+1}\, e^y_{b_m}(k-1) \tag{3.36}$$

and the partial derivative with respect to b_0 is

$$\frac{\partial e^y_{f_m}(k)}{\partial b_0} = \frac{\partial e^y_{f_{m+1}}(k)}{\partial b_0} \tag{3.37}$$

The terminal condition is,

$$e^y_{f_M}(k) = b_0\, e^x_{f_M}(k) \tag{3.38}$$

Taking the partial derivative of equation (3.38) with respect to b_0 yields $e^x_{f_M}(k)$, and the recursive equation to update $b_0(k)$ becomes,

$$b_0(k) = b_0(k-1) - \mu\, e(k)\, e^x_{f_M}(k) \tag{3.39}$$

The gradient estimators for the lattice filter coefficients are scaled by the arbitrary constant b_0, since at the terminal condition $\phi^y_{f_M, ij}(k) = b_0\, \phi^x_{f_m, ij}(k)$ and b_0 is propagated through the calculations. The gradient estimators become,

$$\begin{aligned} \psi_{ij}(k) &= -\, b_0\, e^y_{b_{j-1}}(k-1) \quad i = 1,3 \\ \psi_{ij}(k) &= b_0\, e^x_{b_{j-1}}(k-1) \quad i = 2,4 \end{aligned} \tag{3.40}$$

To test this more general adaptive lattice algorithm the output of a known transfer function with b_0 equal to 0.5 was compared to the output of the ARMA digital lattice filter. The second order transfer function used was,

$$H_f(z) = \frac{0.5 - 0.4\, z^{-1} + 0.89\, z^{-2}}{1 - 0.89\, z^{-1} + 0.25\, z^{-2}} \tag{3.41}$$

The computer generated steady state lattice coefficients are given below and convergence aspects shown in Figure 14.

$b_0 = 0.499946,\quad w_2^1 = -0.120428,\quad w_2^2 = 0.593237,\quad w_1^1 = -0.447423,\quad w_1^2 = 0.166320$

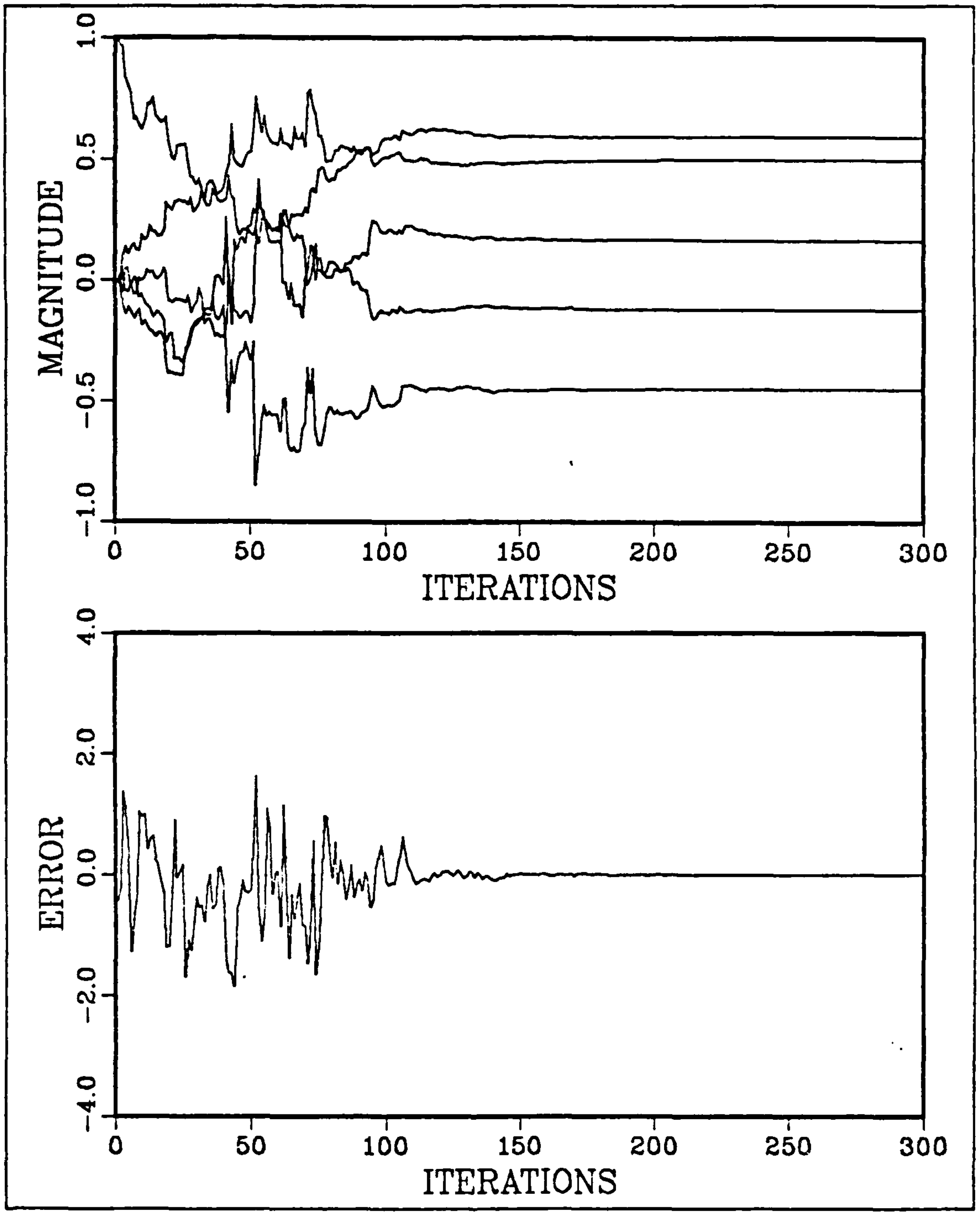

Figure 14. Top: Lattice coefficients. Bottom: Output error.

In order to maintain the same adaptive time constant and misadjustment at each stage in the lattice, the convergence constant is normalized by the power level at each stage [Ref. 15]. Therefore, we can write equations (3.21) and (3.39) as,

$$\begin{aligned} w_i^j(k) &= w_i^j(k-1) - \frac{\mu}{\sigma_j^2(k)}\, e(k)\, \psi_{ij}(k) \\ b_0(k) &= b_0(k-1) - \frac{\mu}{\gamma^2(k)}\, e(k)\, e_{f_M}^x(k) \end{aligned} \tag{3.42}$$

where μ is the convergence constant and $\sigma_j^2(k)$ and $\gamma^2(k)$ are estimates of the power at the j^{th} stage for w_i^j and b_0 respectively and computed as follows:

$$\begin{aligned} \sigma_j^2(k) &= \rho\, \sigma_j^2(k-1) + (1-\rho)\, \psi_{ij}^2(k) \\ \gamma^2(k) &= \rho\, \gamma^2(k-1) + (1-\rho)\left[\, e_{f_M}^x(k)\, \right]^2 \end{aligned} \tag{3.43}$$

Writing equation (3.42) using the notation adopted for the reduced elementary ARMA lattice section we obtain

$$\begin{aligned} r_j(k) &= r_j(k-1) - \frac{\mu}{\sigma_j^2(k)}\, e(k)\, e_{b_{j-1}}^x(k-1) \\ k_j(k) &= k_j(k-1) - \frac{\mu}{\sigma_j^2(k)}\, e(k)\, e_{b_{j-1}}^y(k-1) \\ b_0(k) &= b_0(k-1) - \frac{\mu}{\gamma^2(k)}\, e(k)\, e_{f_M}^x(k) \end{aligned} \tag{3.44}$$

In the above equations ρ is a weighting parameter, $0 \le \rho \le 1$, which distributes the amount of weight given the past power level or current sample. Normalized convergence constants are used in all examples of this thesis. The adaptive lattice algorithm is summarized in Table 1.

In summary, we have derived an adaptive algorithm based on the LMS theory of adaptive coefficient computation. This new adaptive algorithm easily updates the lattice coefficients by using available data. The original requirement to update eight coefficients of an elementary ARMA lattice section was reduced to updating only two coefficients and still being able to describe the lattice. The algorithm is general in that it applies to systems whose terminal condition is an arbitrary constant. The validity of this algorithm was demonstrated through comparisons between hand analysis and computer simulation. In the next chapter, we further demonstrate the convergence of this algorithm.

Table 1. SUMMARY OF ADAPTIVE LATTICE ALGORITHM

With given initial conditions, $e^x_{f_0} = x(k)$, $e^y_{f_0} = y(k)$, $e^x_{b_0}(k-1) = e^x_{b_0}(k-1) = 0$ and all lattice coefficients zero,

Step 1: for k = 1,m compute

$$e^x_{f_m}(k) = e^x_{f_{m-1}}(k) + w_2^m e^x_{b_{m-1}}(k-1) - w_1^m e^y_{b_{m-1}}(k-1)$$
$$e^y_{f_m}(k) = e^y_{f_{m+1}}(k) + w_4^{m+1} e^x_{b_m}(k-1) - w_3^{m+1} e^y_{b_m}(k-1)$$

with output $e^y_{f_0}(k)$

Step 2: for k = 1,m compute

$$e^x_{b_m}(k) = e^x_{b_{m-1}}(k-1) + w_5^m e^x_{f_{m-1}}(k) - w_6^m e^y_{f_{m-1}}(k)$$
$$e^y_{b_m}(k) = e^y_{b_{m-1}}(k-1) - w_7^m e_{f_{m-1}}(k) + w_8^m e^y_{f_{m-1}}(k)$$

Step 3: Update coefficients

$$r_j(k) = r_j(k-1) - \frac{\mu}{\sigma_j^2(k)} e(k)\, e^x_{b_{j-1}}(k-1)$$
$$k_j(k) = k_j(k-1) - \frac{\mu}{\sigma_j^2(k)} e(k)\, e^y_{b_{j-1}}(k-1)$$
$$b_0(k) = b_0(k-1) - \frac{\mu}{\gamma^2(k)} e(k)\, e^x_{f_M}(k)$$

Step 4: Repeat for next iteration i.e. return to step 1.

IV. EXPERIMENTAL RESULTS

The adaptive lattice algorithm derived in Chapter III is now computer simulated to study its convergence performance. The system identification mode of adaptive filtering is considered for this purpose. Figure 15 shows a general system identification configuration. The systems considered are time-invariant and linear. Notice that we apply the same input, white noise in general, to both the reference system and the adaptive lattice filter which is modeling the system. The criterion in this configuration is to minimize the mean-squared error between the system and filter outputs. Thus, in this context, the adaptive algorithm continuously updates the lattice filter parameters in order to minimize the mean-squared error.

The adaptive algorithm is realized as summarized in Table 1. As we mentioned in Chapter III, the two important parameters of the algorithm are the adaptation constant μ and the weighting constant ρ . In what follows, we shall consider convergence studies of both second and third order reference systems (fixed filter transfer functions). Consider the following reference system with transfer function,

$$H_f(z) = \frac{1 + 0.2\,z^{-1} - 0.35\,z^{-2}}{1 - 1.4\,z^{-1} + 0.85\,z^{-2}}$$

This system has complex poles and simple zeros located at $z = (0.7 \pm j0.6)$ and $z = 0.5, -0.7$, respectively. Using a convergence constant $\mu = 0.01$ and power level weighting factor $\rho = 0.45$, the adaptive ARMA digital lattice filter which models the above system has the following steady-state lattice parameters,

$$\begin{array}{rl} \text{terminal condition} & b_0 = 0.999202 \\ \text{lattice coefficients} & r_1^1 = 0.352580 \\ & r_1^2 = -0.174355 \\ & k_1^1 = -0.448228 \\ & k_1^2 = 0.425060 \end{array}$$

Convergence properties of the lattice coefficients and error are shown in Figure 16. The mean-squared error was minimized after approximately 1700 iterations at which time the lattice coefficients reached their steady-state values. When the value of the convergence

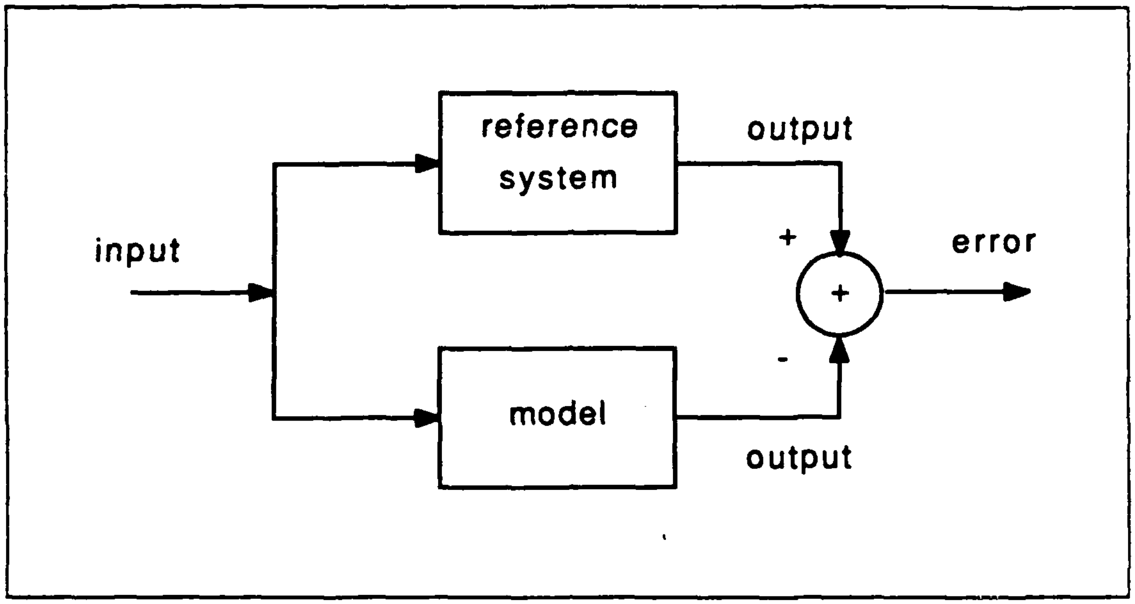

Figure 15. Block diagram of system indentification modeling

constant μ was modestly increased, convergence degraded rapidly. Also, when the weighting factor ρ was increased, convergence deteriorated quickly. From this, we conclude that the convergence constant is the more sensitive input paramcter.

Let us consider another second order dynamic system with transfer function,

$$H_f(z) = \frac{0.5 - 0.2\,z^{-1} + 0.445\,z^{-2}}{1 - z^{-1} + 0.94\,z^{-2}}$$

This system has complex poles at $z = (0.5 \pm j0.8)$ and complex zeros located at $z = (0.2 \pm j0.7)$. Using a convergence constant $\mu = 0.005$ and power level weighting factor $\rho = 0.97$, the adaptive ARMA digital lattice filter which models this system has steady-state lattice parameters as follows,

$$\begin{aligned}
\text{terminal condition } b_0 &= 0.500027 \\
\text{lattice coefficients } r_1^1 &= 0.104654 \\
r_1^2 &= 0.296658 \\
k_1^1 &= -0.429232 \\
k_1^2 &= 0.627718
\end{aligned}$$

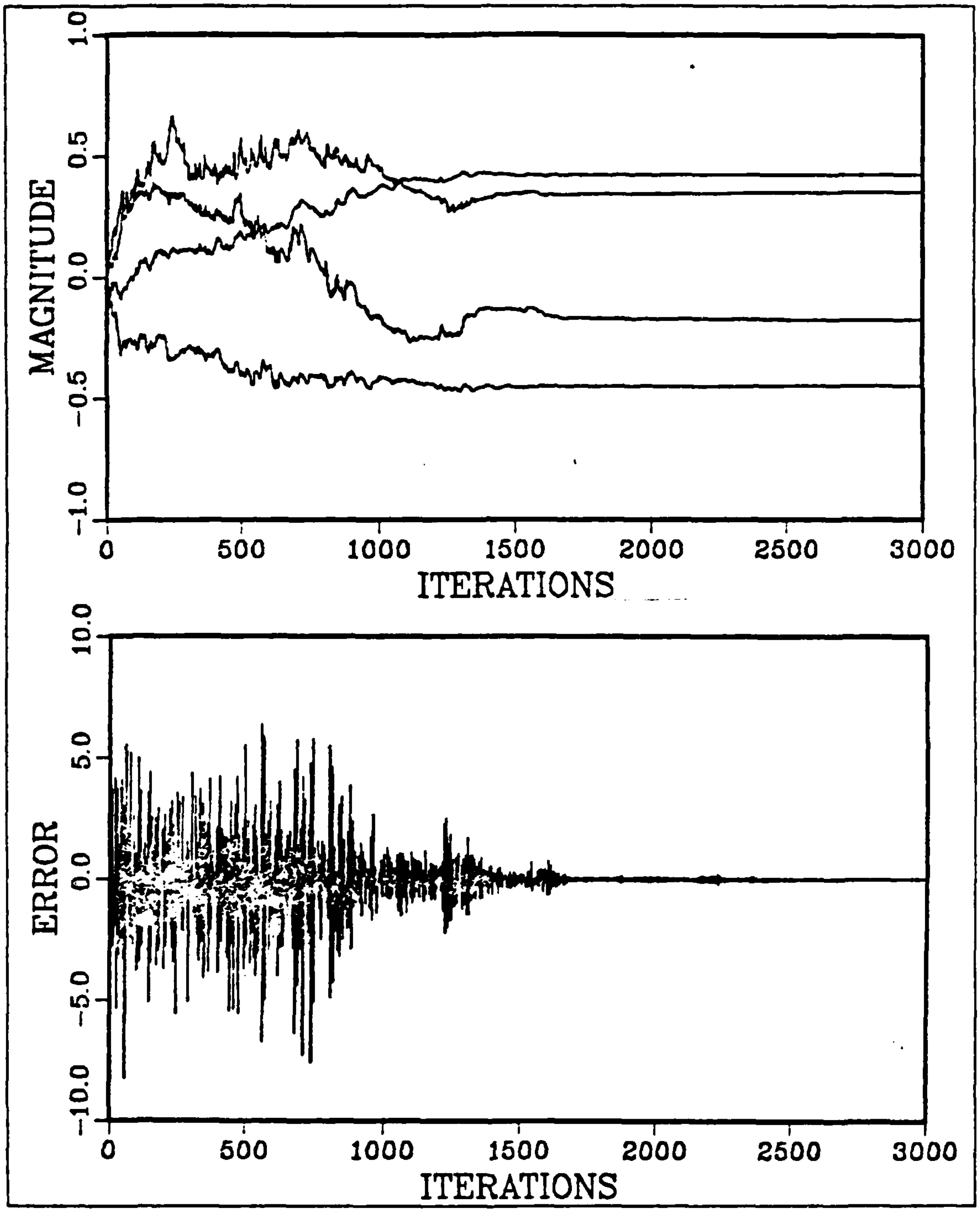

Figure 16. Second order ARMA lattice filter, terminal condition unity.

Convergence properties of this adaptive filter are shown in Figure 17. In this example, convergence was obtained after approximately 2500 iterations. Again, by changing the values of μ and ρ slightly, covergence deteriorated with the convergence constant μ being the more sensitive parameter.

Next we consider a third order reference system with known transfer function,

$$H_f(z) = \frac{0.5 - 0.95\,z^{-1} + 1.33\,z^{-2} - 0.979\,z^{-3}}{1 - 1.69\,z^{-1} + 0.962\,z^{-2} - 0.2\,z^{-3}}$$

The adaptive ARMA digital lattice filter which describes this system has the following steady-state lattice parameters,

$$\begin{aligned}
\text{terminal condition } b_0 &= 0.499970\\
\text{lattice coefficients } r_1^1 &= -0.328447\\
r_1^2 &= 0.399472\\
r_1^3 &= -0.652706\\
k_1^1 &= -0.821738\\
k_1^2 &= -0.091111\\
k_1^3 &= -0.133333
\end{aligned}$$

These parameters were obtained using a convergence constant $\mu = 0.015$ and power level weighting factor $\rho = 0.9$. Convergence properties of this adaptive filter are shown in Figure 18. Steady-state values for the lattice coefficients were obtained after approximately 7100 iterations. It is reasonable to assume that a third order system will converge more slowly than a second order system. The number of iterations required for this third order system to converge is consistant with convergence rates of other adaptive algorithms which model third order systems [Ref. 16]. The input parameter μ was again found to be the more sensitive parameter.

In all the previous examples, the values of μ and ρ may or may not be optimum values. That is, an exhaustive search of all combinations of μ and ρ was not performed to demonstrate convergence of the algorithm. Nevertheless, a number of different ways of realizing the value of the convergence constant μ have been reported in the literature. In one method, Mikhael et. al. [Ref. 17] have obtained a variable μ by using a self optimizing technique. In this method, μ is calculated from the input data as an iteration process and is individually determined for each filter parameter. In another method μ is chosen by using a variable step LMS technique [Ref. 18], where the range of μ is

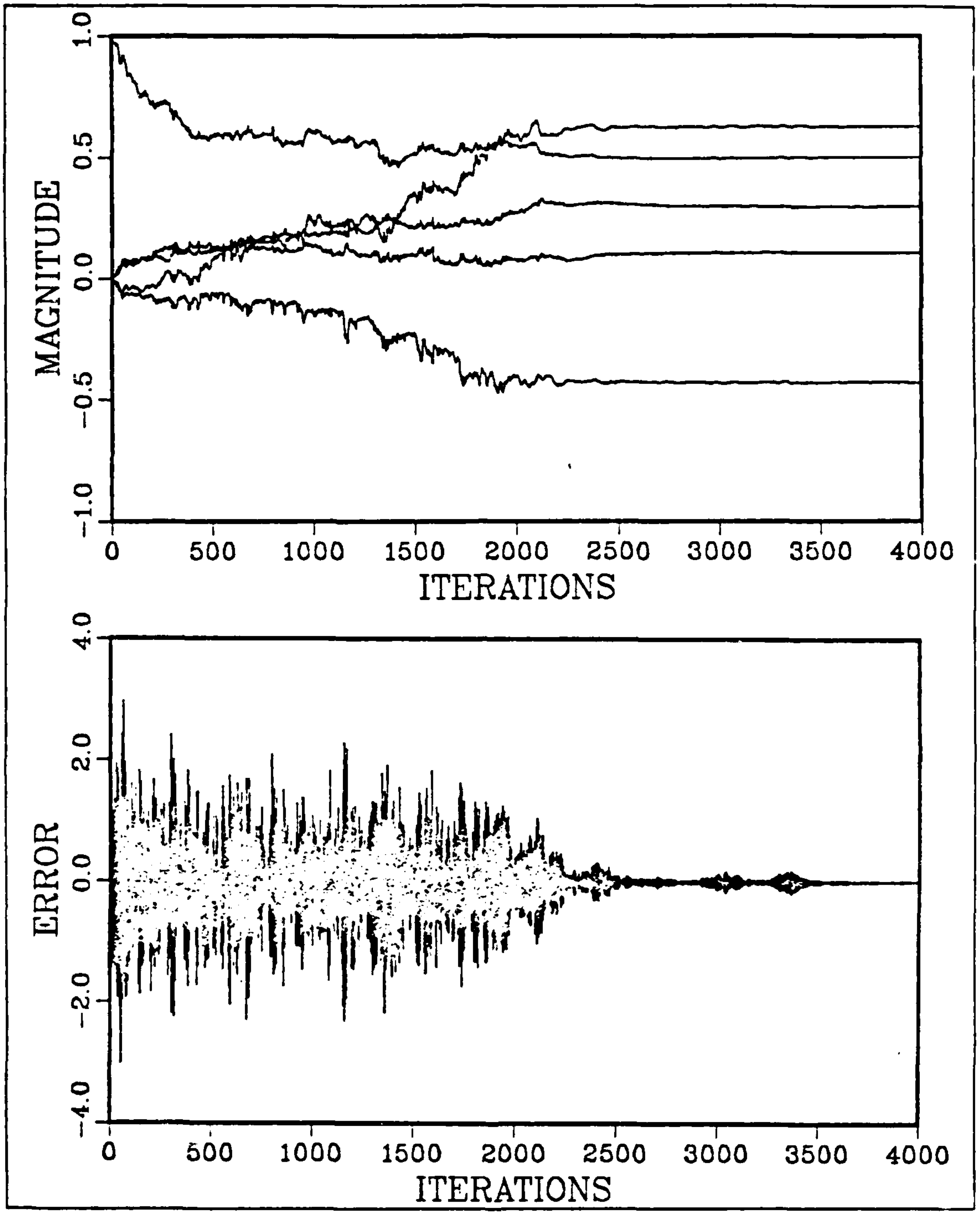

Figure 17. Second order ARMA lattice filter, terminal condition of 0.5.

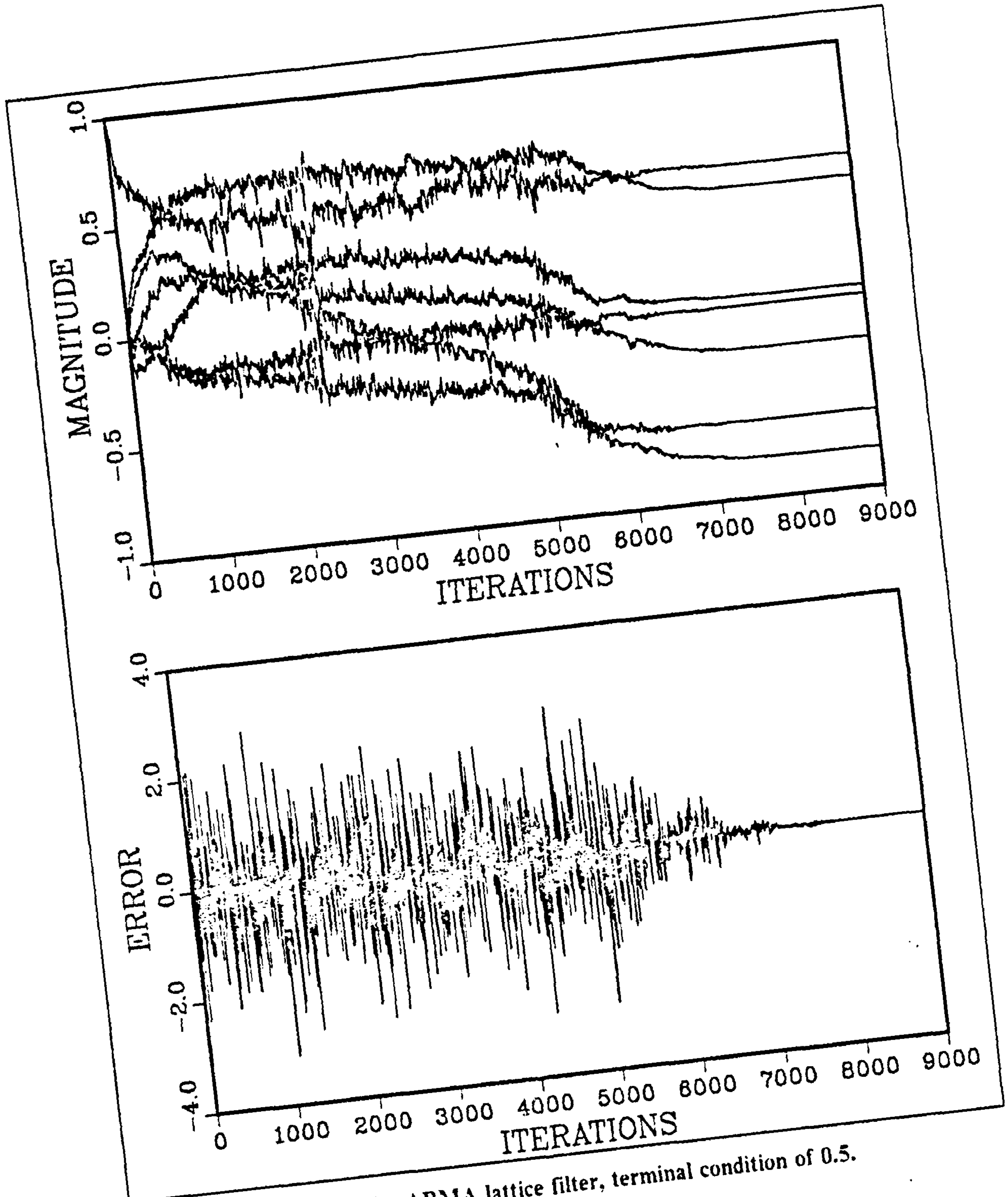

Figure 18. Third order ARMA lattice filter, terminal condition of 0.5.

specified by μ_{max} and μ_{min} which are within the bounds described by equation (3.4) of Chapter III. These techniques of choosing μ during the adaptation process have been shown to improve filter convergence. They, however, require additional computations to achieve this faster convergence. When a combination of the parameters μ and ρ was obtained which yielded convergence, these values were chosen for examples. Besides the examples reported, simulation studies have been carried out for several other cases. In all cases, however, definite convergence of the algorithm has been observed.

In summary, we have demonstrated through computer simulation that the derived adaptive lattice algorithm is suited for system identification modeling. Furthermore, we have shown that there is flexibility in choosing the values of the convergence constant μ and weighting factor ρ . Some techniques for selecting (computing) the value of the convergence constant have been introduced. These methods improve convergence at the cost of additional computations.

A. CONCLUSIONS

In this thesis we have demonstrated that the ARMA parameter estimation algorithm proposed in [Ref. 4: pp. 619-621] is a valid method for obtaining approximations to reference models. Furthermore, the criterion used to derive the algorithm is a generalized form of the Mullis-Roberts criterion for least squares modeling. The AR and MA parameters of the ARMA model can be updated independently as their respective orders increase by one. From the recursive prediction error formulas, an ARMA digital lattice filter was designed with arbitray AR and MA orders.

For the ARMA digital lattice filter, we derived an adaptive lattice algorithm. This algorithm was based on the least mean square method of optimizing coefficients. The derived adaptive lattice algorithm can easily compute the values of the lattice coefficients from available data. The algorithm simplified the number of coefficents required to be updated from eight coefficicients per elementary lattice section to only two such that the filter can be completely described. This savings in computational effort makes the algorithm attractive for identification of unknown systems since many systems require an ARMA model for parsimonious modeling.

Convergence of the adaptive lattice algorithm was demonstrated with several examples in Chapter III. The number of iterations required before convergence varied greatly between second and third order models as well as within second order models. Optimum convergence rates were not sought after as much as proving the convergence of the algorithm. Rapid convergence rates were demonstrated in Chapter II for a second order system upon completion of an extensive search for the optimun values of the convergence constant and power level weighting factor.

Although a method of converting between direct form and lattice realizations for second order ARMA filters was developed, a general algorithm to perform this transformation given the ARMA lattice filter design was not obtained. When solving for a transformation between filter realizations of third order the solution is hindered by nonlinearities.

The objectives of the thesis were sucsessfully accomplished. Some suggestions for future work include the following: (i) extensive theoretical analysis for determining optimum values for μ , (ii) derivation of a generalized algorithm which converts any given ARMA transfer function into a set of lattice parameters, (iii) development of theoretical convergence models for the ARMA adaptive lattice algorithm and analysis of these models and (iv) application of the adaptive lattice filter, both analysis and synthesis forms, in modeling such practical signals as speech. ARMA lattice filter modeling has considerable application potential because of its very accurate modeling of nearly any signal or system of interest.

APPENDIX A. MAIN PROGRAM TO ESTIMATE ARMA PARAMETERS

```
C      THIS PROGRAM COMPUTES THE ARMA PARAMETERS AS THE AR OR MA ORDER
C       OF AN ARMA MODEL INCREASES BY ONE. IT USES THE ARMA PARAMETER
C         ESTIMATION ALGORITHM PROPOSED BY MIYANAGA, NAGAI AND MIKI.
C
C                          VARIABLE DEFINITIONS
C
C      VN    - INPUT VECTOR CONTAINING DATA GENERATED AS WHITE GAUSSIAN
C              NOISE
C      Y     - OUPUT VECTOR OF DIFFERENCE EQUATION WITH VN AS INPUT
C      RX    - AUTOCORRELATION DATA OF INPUT VN
C      RY    - AUTOCORRELATION DATA OF OUTPUT Y
C      RXY   - CROSSCORRELATION DATA OF INPUT AND OUTPUT
C      RYX   - CROSSCORRELATION DATA OF OUTPUT AND INPUT
C      NDATA - NUMBER OF INPUT DATA POINTS
C      KDATA - NUMBER OF INITIAL DATA POINTS TO DISREGARD
C      X1    - INPUT DATA VECTOR AFTER DISCARDING KDATA POINTS
C      Y1    - OUTPUT DATA VECTOR AFTER DISCARDING KDATA POINTS
C      A     - VECTOR CONTAINING CALCULATED AR PARAMETERS
C      B     - VECTOR COINTAINING CALCULATED MA PARAMETERS
C      TXA   - VECTOR CONTAINING COEFFICIENTS FOR FORWARD PREDICTION OF
C              INPUT
C      TYA   - VECTOR CONTAINING COEFFICIENTS FOR FORWARD PREDICTION OF
C              OUPUT
C      TXB   - VECTOR CONTAINING COEFFICIENTS FOR FORWARD PREDICTION OF
C              INPUT
C      TYB   - VECTOR CONTAINING COEFFICIENTS FOR FORWARD PREDICTION OF
C              OUTPUT
C      GA    - VECTOR CONTAINING COEFFICIENTS FOR BACKWARD PREDICTION OF
C              INPUT
C      GB    - VECTOR CONTAINING COEFFICIENTS FOR BACKWARD PREDICTION OF
C              INPUT
C      ZTA   - VECTOR CONTAINING AR COEFFICIENTS FOR BACKWARD PREDICTION
C              OF OUTPUT VALUES
C      ZTB   - VECTOR CONTAINING MA COEFFICIENTS FOR BACKWARD PREDICTION
C              OF OUTPUT VALUES.
C       NI   - LENGTH OF DATA VECTOR X1
C       NK   - LENGTH OF DATA VECTOR X1 MINUS ONE USED TO START
C              CORRELATION COMPUTATIONS.
C       NS   - DESIRED AR ORDER OF ARMA MODEL.
C       NT   - DESIRED MA ORDER OF ARMA MODEL
C       KS   - CURRENT AR ORDER OF UPDATE
C       KT   - CURRENT MA ORDER OF UPDATE
C       VX   - EXPECTED VALUE OF PREDICTION ERROR FOR INPUT SQUARED.
C       VY   - EXPECTED VALUE OF PREDICTION ERROR FOR OUTPUT SQUARED.
C       VXY  - EXPECTED VALUE OF PRODUCT OF PREDICTION ERROR FOR INPUT
```

```
C             AND OUTPUT.
C       VG   - EXPECTED VALUE OF BACKWARD PREDICTION ERROR OF INPUT
C             SQUARED.
C       VZ   - EXPECTED VALUE OF BACKWARD PREDICTION ERROR OF OUTPUT
C             SQUARED.
C       VGZ  - EXPECTED VALUE OF PRODUCT BETWEEN BACKWARD PREDICTION
              ERRORS OF INPUT AND OUTPUT.
C
      DIMENSION VN(-5:2006),Y(-5:2006),RX(0:2000),RY(0:2000),RXY(0:2000)
      DIMENSION RYX(0:2000),X1(2000),Y1(2000),X(12),A(0:21),B(0:21)
      DIMENSION TXA(0:21),TYA(0:21),TXB(0:21),TYB(0:21),GA(0:21)
      DIMENSION GB(0:21),ZTA(0:21),ZTB(0:21)
C
C     INPUT DATA INFORMATION
C
      WRITE (6,1)
1     FORMAT (/' ENTER THE NUMBER OF DATA POINTS:')
      READ (6,*) NDATA
      WRITE (6,2)
2     FORMAT (/' ENTER NUMBER OF INITIAL DATA POINTS TO DISREGARD:')
      READ (6,*) KDATA
C
C     ******************************************************************
C
C     INITIALIZE ARRAYS
C
      DO 10 L=-5,NDATA
      VN(L)=0
      Y(L) =0
10    CONTINUE
      DO 15 L=0,20
      A(L)=0
      B(L)=0
      TXA(L)=0
      TXB(L)=0
      TYA(L)=0
      TYB(L)=0
      GA(L) =0
      GB(L) =0
      ZTA(L)=0
      ZTB(L)=0
15    CONTINUE
      DO 20 L=0,1999
      RX(L)=0
      RY(L)=0
      RXY(L)=0
      RYX(L)=0
20    CONTINUE
      DO 25 L=1,12
```

```
      X(L)=0
25    CONTINUE
C
C     *****************************************************************
C
C     GENERATE WHITE GAUSSIAN NOISE INPUT
C
      ISIZE=NDATA
      IX=152255
      ISORT=0
      MUL=2
      DO 30 K=1,ISIZE
        CALL SRND(IX,X,12,MUL,ISORT)
        XT=-6.0
          DO 35 I=1,12
35        XT=XT+X(I)
        VN(K)=XT
30    CONTINUE
C
C     *****************************************************************
C
C     COMPUTE OUTPUT OF REFERENCE MODEL FILTER AND DISREGARD SPECIFIED
C     NUMBER OF DATA POINTS
C
      DO 40 L=1,NDATA
C     Y(L)=VN(L)+1.6*Y(L-1)-0.95*Y(L-2)
C     Y(L)=VN(L)+0.2*Y(L-1)-0.62*Y(L-2)+0.152*Y(L-3)-0.3016*Y(L-4)
C     Y(L)=VN(L)-0.2*VN(L-1)+0.62*VN(L-2)-0.152*VN(L-3)+0.3016*VN(L-4)
C     Y(L)=VN(L)+0.2*VN(L-1)-0.99*VN(L-2)+0.2*Y(L-1)-0.62*Y(L-2)+0.152*Y
C    &(L-3)-0.3016*Y(L-4)
C     Y(L)=VN(L)-1.6*VN(L-1)+1.45*VN(L-2)+1.2*Y(L-1)-0.72*Y(L-2)
C     Y(L)=VN(L)+0.2*VN(L-1)-0.35*VN(L-2)+1.4*Y(L-1)-0.85*Y(L-2)
      Y(L)=0.5*VN(L)-0.2*VN(L-1)+0.445*VN(L-2)+Y(L-1)-0.94*Y(L-2)

C     Y(L)=VN(L)-2.7*VN(L-1)+3.21*VN(L-2)-1.595*VN(L-3)+1.95*Y(L-1)-1.62
C    &*Y(L-2)+0.54*Y(L-3)
C     Y(L)=VN(L)-1.0*VN(L-1)+0.89*VN(L-2)+0.40*Y(L-1)-0.2121*Y(L-2)-0.20
C    &894*Y(L-3)-1.810373*Y(L-4)
C     Y(L)=0.5*VN(L)-0.95*VN(L-1)+1.33*VN(L-2)-0.979*VN(L-3)+1.69*Y(L-1)
C    &-0.962*Y(L-2)+0.20*Y(L-3)
C     Y(L)=0.5*VN(L)-0.4*VN(L-1)+0.89*VN(L-2)+1.69*Y(L-1)-0.962*Y(L-2)+0
C    &.2*Y(L-3)
C     Y(L)=0.5*VN(L)-0.4*VN(L-1)+0.89*VN(L-2)+0.2*Y(L-1)+0.25*Y(L-2)-0.0
C    &5*Y(L-3)
C     Y(L)=0.5*VN(L)-0.4*VN(L-1)+0.89*VN(L-2)+0.89*Y(L-1)-0.25*Y(L-2)
C     Y(L)=.0154*VN(L)+.0642*VN(L-1)+0.0642*VN(L-2)+0.0154*VN(L-3)+1.99*
C    &Y(L-1)-1.57*Y(L-2)+0.4583*Y(L-3)
C     Y(L)=0.5*VN(L)+0.256*VN(L-1)+0.1234*VN(L-2)+0.0987*VN(L-3)
40    CONTINUE
```

```
      LJ=KDATA+1
      DO 45 L=LJ,NDATA
        LK=L-KDATA
        X1(LK)=VN(L)
        Y1(LK)=Y(L)
45    CONTINUE
C     WRITE (*,77) (Y1(K), K=200,1800)
77    FORMAT (5(1X,F10.6))
C
C     **************************************************************************
C
C     COMPUTE AUTO-CORRELATION AND CROSS-CORRELATION TERMS
C
46    NI=NDATA-KDATA
      NK=NI-1
      CALL CORREL (NI,50,X1,Y1,RX,RY,RXY,RYX,NK)
47    DO 50 L=0,10
      WRITE (*,200) RX(L),RY(L),RXY(L),RYX(L)
      WRITE (9,200) RX(L),RY(L),RXY(L),RYX(L)
      WRITE (9,201)
50    CONTINUE
      WRITE (9,201)
200   FORMAT (2X,4(2X,F14.9))
201   FORMAT (' ')
C
C     **************************************************************************
C
C     INPUT THE DESIRED AR AND MA ORDERS THEN DEFINE INITIAL CONDITIONS
C
48    WRITE (6,3)
3     FORMAT (/' ENTER THE DESIRED AR ORDER:')
      READ (6,*) NS
      WRITE (6,4)
4     FORMAT (/' ENTER THE DESIRED MA ORDER:')
      READ (6,*) NT
C
      KS=0
      KT=0
      A(0)=1.0
      VX=RX(0)
      VY=RY(0)
      VXY=-RYX(0)
      VG=RX(0)
      VZ=RY(0)
      VGZ=-RYX(0)
      TXB(0)=1.0
      TYA(0)=1.0
      GB(0) =1.0
      ZTA(0)=1.0
C
```

```
C     ************************************************************************
C
C     ESTIMATE THE ARMA PARAMETERS
C
300   IF (NT.EQ.0.AND.KS.LT.NS) THEN
         KS=KS+1
      CALL NEWAR(KS,KT,VX,VY,VXY,VG,VZ,VGZ,TXA,TXB,TYA,TYB,GA,GB,ZTA,ZTB
     &,RX,RY,RXY,RYX,A,B)
      GOTO 300
      ELSE
301   IF (NS.EQ.0.AND.KT.LT.NT) THEN
         KT=KT+1
      CALL NEWMA(KS,KT,VX,VY,VXY,VG,VZ,VGZ,TXA,TXB,TYA,TYB,GA,GB,ZTA,ZTB
     &,RX,RY,RXY,RYX,A,B)
      GOTO 301
      ENDIF
      ENDIF
C
      IF (NS.NE.0.OR.NT.NE.0) THEN
302   IF (NS.GE.NT.AND.NT.NE.0.AND.KT.LT.NT) THEN
         KS=KS+1
      CALL NEWAR(KS,KT,VX,VY,VXY,VG,VZ,VGZ,TXA,TXB,TYA,TYB,GA,GB,ZTA,ZTB
     &,RX,RY,RXY,RYX,A,B)
         KT=KT+1
      CALL NEWMA(KS,KT,VX,VY,VXY,VG,VZ,VGZ,TXA,TXB,TYA,TYB,GA,GB,ZTA,ZTB
     &,RX,RY,RXY,RYX,A,B)
      GOTO 302
       ELSE
303      IF (KS.LT.NS) THEN
           KS=KS+1
      CALL NEWAR(KS,KT,VX,VY,VXY,VG,VZ,VGZ,TXA,TXB,TYA,TYB,GA,GB,ZTA,ZTB
     &,RX,RY,RXY,RYX,A,B)
      GOTO 303
         ENDIF
      ENDIF
      ENDIF
C
C     ************************************************************************
C
C     PRINT ESTIMATED ARMA PARAMETERS
C
      WRITE (*,211)
      WRITE (9,211)
      WRITE (*,210) (A(K), K=1,KS)
      WRITE (9,210) (A(K), K=1,KS)
      WRITE (*,211)
      WRITE (9,211)
211   FORMAT(' ')
      WRITE (*,210) (B(K), K=0,KT)
```

```
      WRITE (9,210) (B(K), K=0,KT)
210   FORMAT (' ',1X,4(2X,F13.10))
      STOP
      END
C
C     *****************************************************************
C
C     SUBROUTINE TO COMPUTE CORRELATION TERMS
C
      SUBROUTINE CORREL(N,LAG,X,Y,RX,RY,RXY,RYX,NK1)
      REAL X(0:NK1),Y(0:NK1),RX(0:2000),RY(0:2000),RXY(0:2000)
      REAL RYX(0:2000),SUM1,SUM2,SUM3,SUM4
      DO 70 K=0,LAG
        NJ=N-1-K
        SUM1=0
        SUM2=0
        SUM3=0
        SUM4=0
        ANK=NJ
          DO 60 J=0,NJ
             SUM1=SUM1+X(J+K)*X(J)
             SUM2=SUM2+Y(J+K)*Y(J)
             SUM3=SUM3+X(J+K)*Y(J)
             SUM4=SUM4+X(J)*Y(J+K)
60        CONTINUE
        RX(K)=SUM1/ANK
        RY(K)=SUM2/ANK
        RYX(K)=SUM3/ANK
        RXY(K)=SUM4/ANK
70    CONTINUE
      RETURN
      END
```

APPENDIX B. SUBROUTINE FOR MAIN PROGRAM

```
      SUBROUTINE NEWAR(KS,KT,VX,VY,VXY,VG,VZ,VGZ,TXA,TXB,TYA,TYB,GA,GB,Z
     &TA,ZTB,RX,RY,RXY,RYX,A,B)
C
C     THIS SUBROUTINE COMPUTES AR PARAMETER VALUES FOR AN ARMA MODEL AS
C                    THE AR ORDER INCREASES BY ONE.
C
C                            VARIABLE DEFINITIONS
C
C     KS   - CURRENT AR ORDER
C     KT   - CURRENT MA ORDER
C     RX   - AUTOCORRELATION DATA OF INPUT VN
C     RY   - AUTOCORRELATION DATA OF OUTPUT Y
C     RXY  - CROSSCORRELATION DATA OF INPUT AND OUTPUT
C     RXY  - CROSSCORRELATION DATA OF OUTPUT AND INPUT
C     TXA  - VECTOR CONTAINING AR COEFFICIENTS FOR FORWARD PREDICTION
C            OF INPUT.
C     TXB  - VECTOR CONTAINING MA COEFFICIENTS FOR FORWARD PREDICTION
C            OF INPUT
C     TYA  - VECTOR CONTAINING AR COEFFICIENTS FOR FORWARD PREDICTION
C            OF OUTPUT
C     TYB  - VECTOR CONTAINING MA COEFFICIENTS FOR FORWARD PREDICTION
C            OF OUTPUT
C     GA   - VECTOR CONTAINING AR COEFFICIENTS FOR BACKWARD PREDICTION
C            OF INPUT
C     GB   - VECTOR CONTAINING MA COEFFICIENTS FOR BACKWARD PREDICTION
C            OF INPUT
C     ZTA  - VECTOR CONTAINING AR COEFFICIENTS FOR BACKWARD PREDICTION
C            OF OUTPUT
C     ZTB  - VECTOR CONTAINING MA COEFFICIENTS FOR BACKWARD PREDICTION
C            OF OUTPUT
C     C    - ARRAY WHICH STORES CURRENT VALUES OF TXA
C     D    - ARRAY WHICH STORES CURRENT VALUES OF TYA
C     E    - ARRAY WHICH STORES CURRENT VALUES OF GA
C     F    - ARRAY WHICH STORES CURRENT VALUES OF ZTA
C     P    - ARRAY WHICH STORES CURRENT VALUES OF TXB
C     Q    - ARRAY WHICH STORES CURRENT VALUES OF TYB
C     R    - ARRAY WHICH STORES CURRENT VALUES OF GB
C     S    - ARRAY WHICH STORES CURRENT VALUES OF ZTB
C     A    - VECTOR UPDATED AR COEFFICIENTS OF ARMA MODEL
C     B    - VECTOR CONTAINING UPDATED MA COEFFICIENTS OF ARMA MODEL.
C     TAU1 - CONSTANT COMPUTED FROM CORRELATION DATA AND PREDICTION
C            ERROR COEFFICIENTS.
C     TAU2 - CONSTANT COMPUTED FROM CORRELATION DATA AND PREDICTION
C            ERROR COEFFICIENTS.
C     TAU3 - CONSTANT COMPUTED FROM CORRELATION DATA AND PREDICTION
```

```
C             ERROR COEFFICIENTS.
C      TAU4 - CONSTANT COMPUTED FROM CORRELATION DATA AND PREDICTION
C             ERROR COEFFICIENTS.
C      XMU1 - COEFFICIENT OF AR-TYPE RECURSIVE FORMULA
C      YMU1 - COEFFICIENT OF AR-TYPE RECURSIVE FORMULA
C      MU2  - COEFFICIENT OF AR-TYPE RECURSIVE FORMULA
C      XMU3 - COEFFICIENT OF AR-TYPE RECURSIVE FORMULA
C      YMU3 - COEFFICIENT OF AR-TYPE RECURSIVE FORMULA
C      MU4  - COEFFICIENT OF AR-TYPE RECURSIVE FORMULA
C      MU5  - COEFFICIENT OF AR-TYPE RECURSIVE FORMULA
C      DET  - DETERMINANT OF PREDICTION ERROR MATRIX COMPOSED OF
C             VX,VY,VXY.
C      ERR  - ERROR BETWEEN REFERENCE MODEL OUTPUT AND LATTICE
C             REALIZATION OUTPUT.
C
       DIMENSION RX(0:2000),RY(0:2000),RXY(0:2000),RYX(0:2000),A(0:21)
       DIMENSION B(0:21),TXA(0:21),TYA(0:21),TXB(0:21),TYB(0:21),GA(0:21)
       DIMENSION GB(0:21),ZTA(0:21),ZTB(0:21),C(0:21),D(0:21),E(0:21)
       DIMENSION P(0:21),Q(0:21),R(0:21),S(0:21),F(0:21)
       REAL MU5,MU2,MU4
C
C      COMPUTE VALUES FOR TAU1 THROUGH TAU4
C
       T1S=0
       T2S=0
       T3S=0
       T4S=0
       KI=KS-1
       DO 10 I=0,KI
         T1S=T1S-RYX(I+1)*ZTA(KI-I)
         T2S=T2S+RY(I+1)*ZTA(KI-I)
         T3S=T3S+RY(I+1)*GA(I)
         T4S=T4S+RY(I+1)*ZTA(I)
10     CONTINUE
       T1T=0
       T2T=0
       T3T=0
       T4T=0
       DO 20 J=0,KT
         T1T=T1T+RX(J+1)*ZTB(KT-J)
         T2T=T2T-RXY(J+1)*ZTB(KT-J)
         T3T=T3T-RYX(KI-KT+1+J)*GB(J)
         T4T=T4T-RYX(KI-KT+1+J)*ZTB(J)
20     CONTINUE
       TAU1=T1S+T1T
       TAU2=T2S+T2T
       TAU3=T3S+T3T
       TAU4=T4S+T4T
C
```

```
C     COMPUTE VALUES FOR THE REFLECTION COEFFICIENTS
C
      XMU1=-TAU1/VZ
      YMU1=-TAU2/VZ
      MU2 =-VGZ/VZ
      DET =VX*VY-VXY*VXY
      XMU3=-(VY*TAU1-VXY*TAU2)/DET
      YMU3=-(-VXY*TAU1+VX*TAU2)/DET
      MU4 =(VGZ*TAU4-VZ*TAU3)/(VGZ*VGZ-VG*VZ)
      MU5 =MU4*MU2
C
C     COMPUTE ARMA COEFFICIENTS
C
      DO 16 K=0,KS
      C(K)=TXA(K)
      D(K)=TYA(K)
      E(K)=GA(K)
      F(K)=ZTA(K)
16    CONTINUE
      DO 45 J=1,KS
       TXA(J)=C(J)+XMU1*F(KS-J)
       TYA(J)=D(J)+YMU1*F(KS-J)
       GA(J)=E(J-1)+MU2*F(J-1)
      ZTA(J)=F(J)+XMU3*C(KS-J)+YMU3*D(KS-J)+MU4*E(J-1)+MU5*F(J-1)
45    CONTINUE
      DO 31 K=0,KT
      P(K)=TXB(K)
      Q(K)=TYB(K)
      S(K)=ZTB(K)
      R(K)=GB(K)
31    CONTINUE
      DO 55 J=1,KT
       TXB(J)=P(J)+XMU1*S(KT+1-J)
       TYB(J)=Q(J)+YMU1*S(KT+1-J)
       GB(J) =R(J)+MU2*S(J)
       ZTB(J)=S(J+1)+XMU3*P(KT-J)+YMU3*Q(KT-J)+MU4*R(J)+MU5*S(J)
55    CONTINUE
C     WRITE (*,176) KS
C     WRITE (9,176) KS
176   FORMAT(I2)
C     WRITE (*,175) (ZTB(K), K=0,KT)
C     WRITE (9,175) (ZTB(K), K=0,KT)
175   FORMAT (4(1X,F10.5))
C
C     UPDATE ERRORS
C
650   FORMAT (/' S UPDATE ERROR IS: ',F15.10)
      VX =VX+XMU1*TAU1
      VY =VY+YMU1*TAU2
      VXY=VXY+XMU1*TAU2
```

```
      VG =VG+MU2*VGZ
      VZ=VZ+(XMU3*TAU1+YMU3*TAU2)+MU4*TAU3+MU5*TAU4
      VGZ=TAU3+MU2*TAU4
      ERR=VY-(VXY**2)/VX
      WRITE (*,650) ERR
      WRITE (9,650) ERR
      WRITE (*,888) VX,VY,VG,VZ
888   FORMAT (4(1X,F10.6))
C
C     COMPUTE MODEL COEFFICIENTS
C
      DO 65 J=1,KS
       A(J)=TYA(J)-TXA(J)*VXY/VX
65    CONTINUE
      DO 70 J=1,KT
       B(J)=TYB(J)-TXB(J)*VXY/VX
70    CONTINUE
      B(0)=-VXY/VX
      RETURN
      END
```

APPENDIX C. SUBROUTINE FOR MAIN PROGRAM

```
      SUBROUTINE NEWMA(KS,KT,VX,VY,VXY,VG,VZ,VGZ,TXA,TXB,TYA,TYB,GA,GB,Z
     &TA,ZTB,RX,RY,RXY,RYX,A,B)
C
C     THIS SUBROUTINE COMPUTES MA PARAMETER VALUES FOR AN ARMA MODEL AS
C                    THE MA ORDER INCREASES BY ONE.
C
C                        VARIABLE DEFINITIONS
C
C     KS     - CURRENT AR ORDER
C     KT     - CURRENT MA ORDER
C     RX     - AUTOCORRELATION DATA OF INPUT VN
C     RY     - AUTOCORRELATION DATA OF OUTPUT Y
C     RXY    - CROSSCORRELATION DATA OF INPUT AND OUTPUT
C     RXY    - CROSSCORRELATION DATA OF OUTPUT AND INPUT
C     TXA    - VECTOR CONTAINING AR COEFFICIENTS FOR FORWARD PREDICTION
C              OF INPUT.
C     TXB    - VECTOR CONTAINING MA COEFFICIENTS FOR FORWARD PREDICTION
C              OF INPUT
C     TYA    - VECTOR CONTAINING AR COEFFICIENTS FOR FORWARD PREDICTION
C              OF OUTPUT
C     TYB    - VECTOR CONTAINING MA COEFFICIENTS FOR FORWARD PREDICTION
C              OF OUTPUT
C     GA     - VECTOR CONTAINING AR COEFFICIENTS FOR BACKWARD PREDICTION
C              OF INPUT
C     GB     - VECTOR CONTAINING MA COEFFICIENTS FOR BACKWARD PREDICTION
C              OF INPUT
C     ZTA    - VECTOR CONTAINING AR COEFFICIENTS FOR BACKWARD PREDICTION
C              OF OUTPUT
C     ZTB    - VECTOR CONTAINING MA COEFFICIENTS FOR BACKWARD PREDICTION
C              OF OUTPUT
C     C      - ARRAY WHICH STORES CURRENT VALUES OF TXA
C     D      - ARRAY WHICH STORES CURRENT VALUES OF TYA
C     E      - ARRAY WHICH STORES CURRENT VALUES OF GA
C     F      - ARRAY WHICH STORES CURRENT VALUES OF ZTA
C     P      - ARRAY WHICH STORES CURRENT VALUES OF TXB
C     Q      - ARRAY WHICH STORES CURRENT VALUES OF TYB
C     R      - ARRAY WHICH STORES CURRENT VALUES OF GB
C     S      - ARRAY WHICH STORES CURRENT VALUES OF ZTB
C     A      - VECTOR UPDATED AR COEFFICIENTS OF ARMA MODEL
C     B      - VECTOR CONTAINING UPDATED MA COEFFICIENTS OF ARMA MODEL.
C     TAU1P  - CONSTANT COMPUTED FROM CORRELATION DATA AND PREDICTION
C              ERROR COEFFICIENTS.
C     TAU2P  - CONSTANT COMPUTED FROM CORRELATION DATA AND PREDICTION
C              ERROR COEFFICIENTS.
C     TAU3P  - CONSTANT COMPUTED FROM CORRELATION DATA AND PREDICTION
```

```
C                ERROR COEFFICIENTS.
C       TAU4P - CONSTANT COMPUTED FROM CORRELATION DATA AND PREDICTION
C                ERROR COEFFICIENTS.
C       XETA1 - COEFFICIENT OF MA-TYPE RECURSIVE FORMULA
C       YETA1 - COEFFICIENT OF MA-TYPE RECURSIVE FORMULA
C       ETA2  - COEFFICIENT OF MA-TYPE RECURSIVE FORMULA
C       XETA3 - COEFFICIENT OF MA-TYPE RECURSIVE FORMULA
C       YETA3 - COEFFICIENT OF MA-TYPE RECURSIVE FORMULA
C       ETA4  - COEFFICIENT OF MA-TYPE RECURSIVE FORMULA
C       ETA5  - COEFFICIENT OF MA-TYPE RECURSIVE FORMULA
C       DET   - DETERMINANT OF PREDICTION ERROR MATRIX COMPOSED OF
C                VX,VY,VXY.
C       ERR   - ERROR BETWEEN REFERENCE MODEL OUTPUT AND LATTICE
C
        DIMENSION RX(0:2000),RY(0:2000),RXY(0:2000),RYX(0:2000),A(0:21)
        DIMENSION B(0:21),TXA(0:21),TXB(0:21),TYA(0:21),TYB(0:21),GA(0:21)
        DIMENSION GB(0:21),ZTA(0:21),ZTB(0:21),C(0:21),D(0:21)
        DIMENSION E(0:21),F(0:21),P(0:21),Q(0:21),R(0:21),S(0:21)
C
C       COMPUTE VALUES FOR TAU1 PRIME THROUGH TAU4 PRIME
C
        T1TP=0
        T2TP=0
        T3TP=0
        T4TP=0
        KJ=KT-1
        DO 10 I=0,KJ
         T1TP=T1TP+RX(I+1)*GB(KJ-I)
         T2TP=T2TP-RXY(I+1)*GB(KJ-I)
         T3TP=T3TP+RX(I+1)*ZTB(I)
         T4TP=T4TP+RX(I+1)*GB(I)
10      CONTINUE
        T1SP=0
        T2SP=0
        T3SP=0
        T4SP=0
        DO 20 J=0,KS
         T1SP=T1SP-RYX(J+1)*GA(KS-J)
         T2SP=T2SP+RY(J+1)*GA(KS-J)
         T3SP=T3SP-RXY(KJ-KS+1+J)*ZTA(J)
         T4SP=T4SP-RXY(KJ-KS+1+J)*GA(J)
20      CONTINUE
        TAU1P=T1TP+T1SP
        TAU2P=T2TP+T2SP
        TAU3P=T3TP+T3SP
        TAU4P=T4TP+T4SP
C
C       COMPUTE VALUES FOR REFLECTION COEFFICIENTS
C
        XETA1=-TAU1P/VG
```

```
      YETA1=-TAU2P/VG
      ETA2 =-VGZ/VG
      DET  =VX*VY-VXY*VXY
      XETA3=-(VY*TAU1P-VXY*TAU2P)/DET
      YETA3=-(-VXY*TAU1P+VX*TAU2P)/DET
      ETA4 =(VGZ*TAU4P-VG*TAU3P)/(VGZ*VGZ-VG*VZ)
      ETA5 =ETA4*ETA2
C
C     COMPUTE ARMA PARAMETERS
C
      DO 16 K=0,KT
      P(K)=TXB(K)
      Q(K)=TYB(K)
      R(K)=GB(K)
      S(K)=ZTB(K)
16    CONTINUE
165   FORMAT (4(1X,F10.5))
      DO 45 J=1,KT
       TXB(J)=P(J)+XETA1*R(KT-J)
       TYB(J)=Q(J)+YETA1*R(KT-J)
       GB(J)=R(J)+XETA3*P(KT-J)+YETA3*Q(KT-J)+ETA4*S(J-1)+ETA5*R(J-1)
       ZTB(J)=S(J-1)+ETA2*R(J-1)
45    CONTINUE
      DO 41 K=0,KS
      C(K)=TXA(K)
      D(K)=TYA(K)
      E(K)=GA(K)
      F(K)=ZTA(K)
41    CONTINUE
175   FORMAT (5(1X,F10.5))
      DO 55 J=1,KS
       TXA(J)=C(J)+XETA1*E(KS+1-J)
       TYA(J)=D(J)+YETA1*E(KS+1-J)
       GA(J)=E(J+1)+XETA3*C(KS-J)+YETA3*D(KS-J)+ETA4*F(J)+ETA5*E(J)
       ZTA(J)=F(J)+ETA2*E(J)
55    CONTINUE
C
C     UPDATE ERRORS
C
      VX=VX+XETA1*TAU1P
      VY=VY+YETA1*TAU2P
      VXY=VXY+XETA1*TAU2P
      VG=VG+(TAU1P*XETA3+TAU2P*YETA3)+ETA4*TAU3P+ETA5*TAU4P
      VZ=VZ+ETA2*VGZ
      VGZ=TAU3P+ETA2*TAU4P
      ERR=VY-(VXY**2)/VX
      WRITE (*,66) ERR
      WRITE (9,66) ERR
66    FORMAT (/' T UPDATE ERROR IS: ',F15.10)
```

```
      WRITE (*,889) VX,VY,VG,VZ
889   FORMAT (4(1X,F10.6))
C
C     COMPUTE MODEL COEFFICIENTS
C
      DO 65 J=1,KT
       B(J)=TYB(J)-TXB(J)*VXY/VX
65    CONTINUE
      DO 70 J=1,KS
       A(J)=TYA(J)-TXA(J)*VXY/VX
70    CONTINUE
      B(0)=-VXY/VX
      RETURN
      END
```

APPENDIX D. ADAPTIVE LATTICE ALGORITHM PROGRAM

```
C     THIS PROGRAM CALCULATES VALUES OF THE LATTICE COEFFICENTSAND
C      OUTPUT OF AN ARMA DIGITAL LATTICE FILTER USING AN ADAPTIVE
C                        LATTICE ALGORITHM.
C
C                       VARIABLE DEFINITIONS
C
C     X      - ARRAY OF INPUT DATA, COMPUTER GENERATED WHITE GAUSSIAN
C              NOISE WITH UNIT VARIANCE.
C     AK     - ARRAY OF LATTICE COEFFICIENTS.
C     R      - ARRAY OF LATTICE COEFFICIENTS.
C     BO     - TERMINAL CONDITION OF LATTICE REALIZATION.
C     M      - NUMBER OF LATTICE STAGES (EQUIVALENT TO ORDER OF ARMA
C              MODEL).
C     EXF    - ARRAY OF FORWARD PREDICTION ERRORS FOR INPUT X.
C     EXB    - ARRAY OF BACKWARD PREDICTION ERRORS FOR INPUT X.
C     EXBD   - ARRAY OF DELAYED BACKWARD PREDICTION ERRORS FOR INPUT X.
C     EYF    - ARRAY OF FORWARD PREDICTION ERRORS FOR OUTPUT Y.
C     EYB    - ARRAY OF BACKWARD PREDICTION ERRORS FOR OUTPUT Y.
C     EYBD   - ARRAY OF DELAYED BACKWARD PREDICTION ERRORS FOR OUTPUT Y.
C     ERROR  - DIFFERENCE BETWEEN REFERENCE MODEL OUTPUT AND LATTICE
C              REALIZATION OUTPUT.
C     YE     - ARRAYS CONTAINING LATTICE COEFFICIENT VALUES AT EACH
C              ITERATION.
C     MU     - CONVERGENCE CONSTANT.
C     RHO    - WEIGHT GIVEN TO CUURENT POWER LEVEL AT EACH STAGE OF THE
C              LATTICE STRUCTURE.
C     SIGK   - POWER LEVEL USED TO NORMALIZE CONVERGENCE CONSTANT WHEN
C              UPDATING AK LATTICE COEFFICIENTS.
C     SIGR   - POWER LEVEL USED TO NORMALIZE CONVERGENCE CONSTANT WHEN
C              UPDATING R LATTICE COEFFICIENTS.
C     SIGB   - POWER LEVEL USED TO NORMALIZE CONVERGENCE CONSTANT WHEN
C              UPDATING TERMINAL CONDITION BO.
C
      DIMENSION EXF(10),EXB(10),EXBD(10),EYF(10),EYB(10),EYBD(10),R(10)
      DIMENSION AK(10), X(9900), ERROR(9900), V(12), YE(6,9900)
      REAL MU
      SIGK=1.
      SIGR=1.
      SIGB=1.
C
C     INITIALIZE ARRAYS
C
      DO 5 I=1,10
      EXF(I)=0
      EXB(I)=0
```

```
      EXBD(I)=0
      EYF(I)=0
      EYB(I)=0
      EYBD(I)=0
      R(I)=0
      AK(I)=0
5     CONTINUE
      DO 6 I=1,9000
      X(I)=0
      ERROR(I)=0
6     CONTINUE
C
C     ENTER VALUE OF THE CONVERGENCE CONSTANT MU AND VALUE OF RHO.
C
      M=2
      N=300
      WRITE (6,*) 'ENTER MU'
      READ(6,*) MU
      WRITE (6,*) 'ENTER RHO'
      READ (6,*) RHO
C
C     GENERATE WHITE GAUSSIAN NOISE
C
      ISIZE = N
      IX = 152255
      ISORT = 0
      MUL = 2
      DO 7 K= 1,ISIZE
       CALL SRND(IX,V,12,MUL,ISORT)
       XT=-6.0
         DO 8 I=1,12
8        XT=XT+V(I)
       X(K)=XT
7     CONTINUE
C
C     COMPUTE OUTPUT OF REFERENCE MODEL FILTER AND LATTICE STRUCTURE
C     THEN COMPUTE THE ERROR.
C
C     REFERENCE MODEL

      Y3=0
      Y2=0
      Y1=0
      X3=0
      X2=0
      X1=0
      B0=1.
      DO 100 I=1,N
C     YF=X(I)-0.8*X1+1.78*X2+0.89*Y1-0.25*Y2
```

```
      YF=0.5*X(I)-0.4*X1+.89*X2+0.89*Y1-0.25*Y2
C     YF=X(I)-2.7*X1+3.21*X2-1.595*X3+1.95*Y1-1.62*Y2+0.54*Y3
C     YF=0.5*X(I)-0.95*X1+1.33*X2-0.979*X3+1.69*Y1-0.962*Y2+0.2*Y3
C     YF=X(I)+0.2*X1-0.35*X2+1.4*Y1-0.85*Y2
C     YF=0.5*X(I)-0.2*X1+0.445*X2+1.0*Y1-0.94*Y2

C     Y3=Y2
      Y2=Y1
      Y1=YF
C     X3=X2
      X2=X1
      X1=X(I)
C
C     LATTICE FILTER
C
      EXF(1)=X(I)
      EXB(1)=X(I)
      DO 10 K=1,M
10    EXF(K+1)=EXF(K)+R(K)*EXBD(K)-AK(K)*EYBD(K)
      EYF(M+1)=B0*EXF(M+1)
      DO 20 K=1,M
20    EYF(M+1-K)=EYF(M+2-K)+R(M+1-K)*EXBD(M+1-K)-AK(M+1-K)*EYBD(M+1-K)
      EYB(1)=EYF(1)
      DO 30 K=1,M-1
      EXB(K+1)=EXBD(K)+R(K)*EXF(K)-R(K)*EYF(K)
30    EYB(K+1)=EYBD(K)+AK(K)*EYF(K)-AK(K)*EXF(K)
      YL=EYF(1)
      ERROR(I)=YF-YL
      ERR=YF-YL
      CALL UPDATE (R,AK,EYBD,EXBD,ERR,MU,M,SIGK,SIGR,B0,RHO)
      CSB=EXF(M+1)*EXF(M+1)
      SIGB=RHO*SIGB+(1-RHO)*CSB
      B0=B0+(MU/SIGB)*ERR*EXF(M+1)
      DO 40 K=1,M
      EXBD(K)=EXB(K)
40    EYBD(K)=EYB(K)
      DO 50 J=1,M
      YE(J,I)=AK(J)
50    YE(J+M,I)=R(J)
202   FORMAT (2(1X,F10.6))
100   CONTINUE
C
C     PRINT THE ERROR AND VALUES OF THE LATTICE COEFFICIENTS.
C
      WRITE (*,200) (ERROR(K), K=1,N,10)
      WRITE (9,200) (ERROR(K), K=1,N,10)
200   FORMAT (5(1X,F10.6))
      WRITE (9,209)
209   FORMAT(' ')
      WRITE (*,201) (R(K), K=1,M)
```

```
      WRITE (*,201) (AK(K), K=1,M)
      WRITE (9,201) (R(K), K=1,M)
      WRITE (9,201) (AK(K), K=1,M)
      WRITE (*,205) BO
      WRITE (9,205) BO
205   FORMAT (F10.6)
201   FORMAT (5(1X,F10.6))
C
C     CALL PLOTTING ROUTINES TO PLOT ERROR AND LATTICE COEFFICIENTS.
C
      CALL PLOT (ERROR,N)
C     CALL PLOT1 (YE,N)
      STOP
      END
C
C     SUBROUTINE WHICH UPDATES LATTICE COEFFICIENTS.
C
      SUBROUTINE UPDATE(R,AK,EYBD,EXBD,ERR,MU,M,SIGK,SIGR,BO,RHO)
      DIMENSION R(10),AK(10),EYBD(10),EXBD(10)
      REAL MU
      CSK=0.
      CSR=0.
      DO 20 J=1,M
      CSK=CSK+EYBD(J)*EYBD(J)*BO**2
20    CSR=CSR+EXBD(J)*EXBD(J)*BO**2
      SIGK=RHO*SIGK+(1-RHO)*CSK
      SIGR=RHO*SIGR+(1-RHO)*CSR
      DO 10 J=1,M
      R(J)=R(J)+(MU/SIGR)*ERR*EXBD(J)*BO
      AK(J)=AK(J)-(MU/SIGK)*ERR*EYBD(J)*BO
10    CONTINUE
      RETURN
      END
C
C     PLOTTING ROUTINE TO PLOT ERROR
C
      SUBROUTINE PLOT(Y,N)
      DIMENSION Y(N),X(9900)
      DO 10 J=1,N
10    X(J)=J
      CALL TEK618
C     CALL PRTPLT(72,6)
C     CALL SHERPA('ADAPTIVE','A',3)
      CALL RESET('ALL')
      CALL PAGE(8.50,6.0)
      CALL HWROT('AUTO')
      CALL XINTAX
      CALL AREA2D(5.0,3.0)
      CALL HEIGHT(0.14)
      CALL COMPLX
```

```
      CALL SHDCHR(90.0,1,0.002,1)
      CALL HEADIN('LEARNING CURVES$',100,2.0,1)
      CALL XNAME('ITERATIONS$',100)
      CALL YNAME('ERROR$',100)
C     CALL MESSAG('  ADAPTIVE FILTER   $',100,3.0,-0.8)
      CALL THKFRM(0.03)
      CALL FRAME
      CALL GRAF(0,'SCALE',N,-3.00,'SCALE',3.00)
C     CALL THKCRV(0.02)
      CALL CURVE(X,Y,N,0)
      CALL ENDPL(0)
      CALL DONEPL
      RETURN
      END
C
C     SUBROUTINE TO PLOT LATTICE COEFFICIENTS
C
      SUBROUTINE PLOT1 (YE,N)
      DIMENSION YE(6,9900),X(9900),Y(9900),YD(9900),A(10)
C.....TRUE VALUES OF THE PARAMETERS
C     A(1)=-0.240719
C     A(2)=0.125
C     A(3)=-0.195719
C     A(4)=0.8900
C     A(5)=0.89
      DO 10 J=1,N
10    X(J)=J
      CALL TEK618
C     CALL PRTPLT(72,6)
C     CALL SHERPA('MENNECKE','A',3)
C.....PRINT SHERPA FILE: SHERPA XXYYZZXX SHGRAPH A
      CALL RESET('ALL')
      CALL PAGE(8.50,6.0)
      CALL HWROT('AUTO')
      CALL XINTAX
      CALL AREA2D(5.0,3.0)
      CALL HEIGHT(0.14)
      CALL COMPLX
      CALL SHDCHR(90.0,1,0.002,1)
      CALL HEADIN('PARAMETERS$',100,2.0,1)
      CALL XNAME('ITERATIONS$',100)
      CALL YNAME('MAGNITUDE$',100)
C     CALL MESSAG('ADAPTIVE ARMA LATTICE$',100,3.0,-0.8)
      CALL THKFRM(0.03)
      CALL FRAME
      CALL GRAF(0,'SCALE',N,-1.00,'SCALE',1.0)
C     CALL THKCRV(0.02)
C.....TO PLOT ESTIMATES
      DO 20 K=1,4
```

```
      DO 30 J=1,N
30    Y(J)=YE(K,J)
      CALL CURVE(X,Y,N,0)
20    CONTINUE
C.....TO PLOT TRUE PARAMETERS
      DO 40 K=1,4
      DO 50 J=1,N
50    Y(J)=A(K)
      CALL DASH
      CALL CURVE(X,Y,N,0)
40    CONTINUE
      CALL ENDPL(0)
      CALL DONEPL
      RETURN
      END
```

APPENDIX E. DERIVATION OF DIFFERENCE EQUATION FOR TWO STAGE ARMA LATTICE FILTER

The lattice filter is described by the expressions for the forward and backward prediction errors, equations (3.6) and (3.7) respectively,

$$
\begin{aligned}
e^x_{f_1}(k) &= x(k) + w^1_2\, x(k-1) - w^1_1\, y(k-1) \\
e^x_{f_2}(k) &= e^x_{f_1}(k) + w^2_2\, e^x_{b_1}(k-1) - w^2_1\, e^y_{b_1}(k-1) \\
e^x_{b_1}(k) &= x(k-1) + w^1_2\, x(k) - w^1_4\, y(k) \qquad\qquad (E-1) \\
e^y_{b_1}(k) &= y(k-1) - w^1_1\, x(k) + w^1_3\, y(k) \\
e^y_{f_1}(k) &= e^x_{f_2}(k) + w^2_4\, e^x_{b_1}(k-1) - w^2_3\, e^y_{b_1}(k-1)
\end{aligned}
$$

and the expression for the filter output,

$$
y(k) = e^y_{f_1}(k) + w^1_4\, x(k-1) - w^1_3\, y(k-1) \qquad (E-2)
$$

Substituting for $e^y_{f_1}(k)$ in equation (E-2) yields,

$$
y(k) = e^x_{f_2}(k) + w^2_4\, e^x_{b_1}(k-1) - w^2_3\, e^y_{b_1}(k-1) + w^1_4\, x(k-1) - w^1_3\, y(k-1) \quad (E-3)
$$

Substituting for $e^x_{b_1}(k-1)$ and $e^y_{b_1}(k-1)$ in equation (E-3)

$$
\begin{aligned}
y(k) = e^x_{f_2}(k) &+ w^2_4 \left[\, x(k-2) + w^1_2\, x(k-1) - w^1_4\, y(k-1) \,\right] \\
&- w^2_3 \left[\, y(k-2) - w^1_1\, x(k-1) + w^1_3\, y(k-1) \,\right] \qquad (E-4) \\
&+ w^1_4\, x(k-1) - w^1_3\, y(k-1)
\end{aligned}
$$

Substituting for $e^x_{f_2}(k)$ in (E-4) we obtain

$$
\begin{aligned}
y(k) = e^x_{f_1}(k) &+ w^2_2\, e^x_{b_1}(k-1) - w^2_1\, e^y_{b_1}(k-1) + w^2_4\, x(k-2) + w^2_4\, w^1_2\, x(k-1) \\
&- w^2_4\, w^1_4\, y(k-1) - w^2_3\, y(k-2) + w^2_3\, w^1_1\, x(k-1) - w^2_3\, w^1_3\, y(k-1) \quad (E-5) \\
&+ w^1_4\, x(k-1) - w^1_3\, y(k-1)
\end{aligned}
$$

From (E-1), substitute for $e^x_{b_1}(k-1)$ and $e^y_{b_1}(k-1)$ in (E-5) to obtain

$$
\begin{aligned}
y(k) = e^x_{f_1}(k) &+ w_2^2 \left[x(k-2) + w_2^1\, x(k-1) - w_4^1\, y(k-1) \right] \\
&- w_1^2 \left[y(k-2) - w_1^1\, x(k-1) + w_3^1\, y(k-1) \right] \\
&+ w_4^2\, x(k-2) + w_4^2\, w_2^1\, x(k-1) - w_4^2\, w_4^1\, y(k-1) \\
&- w_3^2\, y(k-2) + w_3^2\, w_1^1\, x(k-1) - w_3^2\, w_3^1\, y(k-1) \\
&+ w_4^1\, x(k-1) - w_3^1\, y(k-1)
\end{aligned} \qquad (E-6)
$$

Now, from (E-1), substituting for $e^x_{f_1}(k)$ in (E-6) yields

$$
\begin{aligned}
y(k) = x(k) &+ w_2^1\, x(k-1) - w_1^1\, y(k-1) + w_2^2\, x(k-2) + w_2^2\, w_2^1\, x(k-1) \\
&- w_2^2\, w_4^1\, y(k-1) - w_1^2\, y(k-2) + w_1^2\, w_1^1\, x(k-1) - w_1^2\, w_3^1\, y(k-1) \\
&+ w_4^2\, x(k-2) + w_4^2\, w_2^1\, x(k-1) - w_4^2\, w_4^1\, y(k-1) - w_3^2\, y(k-2) \\
&+ w_3^2\, w_1^1\, x(k-1) - w_3^2\, w_3^1\, y(k-1) + w_4^1\, x(k-1) - w_3^1\, y(k-1)
\end{aligned} \qquad (E-7)
$$

Grouping terms we get,

$$
\begin{aligned}
y(k) = x(k) &+ (w_2^1 + w_2^1\, w_2^2 + w_1^1\, w_1^2 + w_2^1\, w_4^2 + w_1^1\, w_3^2 + w_4^1)\, x(k-1) \\
&+ (w_2^2 + w_4^2)\, x(k-2) \\
&- (w_1^1 + w_4^1\, w_2^2 + w_3^1\, w_1^2 + w_4^1\, w_4^2 + w_3^1\, w_3^2 + w_3^1)\, y(k-1) \\
&- (w_1^2 + w_3^2)\, y(k-2)
\end{aligned} \qquad (E-8)
$$

From the gradient estimator and coefficient update equations we know that the following relationships among lattice coefficients are true

$$
w_1^1 = w_3^1 \quad w_1^2 = w_3^2 \quad w_2^1 = w_4^1 \quad w_2^2 = w_4^2 \qquad (E-9)
$$

Using these equalities in equation (E-8), we obtain the final expression for the difference equation,

$$
\begin{aligned}
y(k) = x(k) &+ 2(w_2^1 + w_1^1\, w_1^2 + w_2^1\, w_2^2)\, x(k-1) + 2\, w_2^2\, x(k-2) \\
&- 2(w_1^1 + w_2^1\, w_2^2 + w_1^1\, w_1^2)\, y(k-1) - 2\, w_1^2\, y(k-2)
\end{aligned} \qquad (E-10)
$$

LIST OF REFERENCES

1. Friedlander, B., "Lattice filters for adaptive processing," *Proc. IEEE*, vol. 70, no. 8, pp. 829-867, August 1982.

2. Johnson, C. R., Jr., "Adaptive IIR filtering: Current results and open issues," *IEEE Trans. Inform. Theory*, vol. IT-30, pp. 237-250, March 1984.

3. Karlsson, E. and Monson, H., "Least squares ARMA modeling of linear time-varying systems: Lattice filter structures and fast RLS algorithms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, no. 7, pp. 994-1014, July 1987.

4. Miyanaga, Y.,Nagai, N. and Miki, N., "ARMA digital lattice filter based on new criterion," *IEEE Trans. Circuits Syst.*, vol. CAS-34, no. 6, June 1987.

5. Mullis, C. T. and Roberts, R. A., "The use of second-order information in the approximation of discrete-time linear systems," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, no. 3, pp. 226-238, June 1976.

6. Orfanidis, S. J., *Optimum Signal Processing, an Introduction*, Macmillan, 1985.

7. Friedlander, B., "Lattice methods for spectral estimation," *Proc. IEEE*, vol. 70, no. 9, pp. 990-1017, September 1982.

8. Markel, J. D. and Gray A. H., Jr., *Linear Prediction of Speech*, pp. 92-97, New York: Springer-Verlag, 1976.

9. Haykin, S., *Adaptive Filter Theory*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1986.

10. Oppenheim, A. V. and Schafer, R.W., *Digital Signal Processing*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1975.

11. Ljung, L., *System Identification: Theory for the User*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1987.

12. Goodwin, G. C. and Sin, K. S., *Adaptive Filtering, Prediction and Control*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1984.

13. Widrow, B. and Hoff, M. E., *Adaptive Switching Circuits*, IRE Wescon Convention Record, Pt. 4, 1960.

14. Widrow, B. and Stearns, S. D., *Adaptive Signal Processing*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1985.

15. Griffiths, L. T., "An adaptive lattice structure for noise-cancelling applications", *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp. 87-90, Tulsa, Oklahoma, April 1978.

16. Parikh, D., Ahmed, N. and Stearns, S. D., "An adaptive lattice algortihm for recursive filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, no. 1, pp. 110-111, February 1980.

17. Mikhael, W. B., Wu, F. H., Kazovsky, L. G., Kang, G. S. and Fransen, L. J., "Adaptive filters with individual adaptation of parameters," *IEEE Trans. Circuits Syst.*, vol. CAS-33, pp. 677-686, July 1986.

18. Harris, R. W., Chalaries, D. M. and Bishop, F. A., "A variable step (vs) adaptive filter algorithm," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 309-316, April 1986.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Department Chairman Electrical and Computer Engineering Department Naval Postgraduate School Monterey, CA 93940-5000	1
4.	Dr. Murali Tummala Electrical and Computer Engineering Department Naval Postgraduate School Monterey, CA 93940-5000	2
5.	Dr. Roberto Cristi Electrical and Computer Engineering Department Naval Postgraduate School Monterey, CA 93940-5000	1
6.	Dr. Rabi N. Madan Office of Naval Research 800 North Quincy Street Arlington, VA 22217-5000	1
7.	Commander Naval Ocean Systems Command Attn: Dr. Eugene P. Cooper, Code 013 San Diego, CA 92152-5000	1
8.	Surface Warfare Officers School Command Attn: Lt. Donald W. Mennecke Department Head Course Newport, RI 02841	1